INFORMATION AND COMPUTATION **101**, 150–201 (1992)

Probabilistic Logic Programming

RAYMOND NG AND V. S. SUBRAHMANIAN

Department of Computer Science, A. V. Williams Building, University of Maryland, College Park, Maryland 20742

Of all scientific investigations into reasoning with uncertainty and chance, probability theory is perhaps the best understood paradigm. Nevertheless, all studies conducted thus far into the semantics of quantitative logic programming have restricted themselves to non-probabilistic semantic characterizations. In this paper, we take a few steps towards rectifying this situation. We define a logic programming language that is syntactically similar to the annotated logics of Blair and Subrahmanian (*Theoret. Comput. Sci.* **68** (1987), 35–54; *J. Non-Classical Logic* **5** (1988), 45–73) but in which the truth values are interpreted probabilistically. A probabilistic model theory and fixpoint theory is developed for such programs. This probabilistic model theory satisfies the requirements proposed by Fenstad (*in* "Studies in Inductive Logic and Probabilities" (R. C. Jeffrey, Ed.), Vol. 2, pp. 251–262, Univ. of California Press, Berkeley, 1980) for a function to be called probabilistic. The logical treatment of probabilities is complicated by two facts: first, that the connectives cannot be interpreted truth-functionally when truth values are regarded as probabilities; second, that negation-free definite-clause-like sentences can be inconsistent when interpreted probabilistically. We address these issues here and propose a formalism for probabilistic reasoning in logic programming. To our knowledge, this is the first probabilistic characterization of logic programming semantics.

1. INTRODUCTION

Probabilities play a central role in our understanding of the world, and in the way in which we reason about the world. For instance, it is not uncommon to hear, from the Department of Health, that 94% of all people who test positive on the only existing test (called HIV) for AIDS actually have AIDS.

Suppose now that an insurance company is considering an application from John Doe for health insurance. John Doe had a positive HIV test. Thus, there is a 6% chance that John Doe does not have AIDS. Nevertheless, it is unlikely that the insurance company will insure John Doe.

This scenario has been described to demonstrate that reasoning about probabilistic and statistical information is common in many real life

0890-5401/92 $5.00

situations (for numerous examples on the applications of probability theory to human reasoning, see Gnedenko and Khinchin, 1962). Often, such probabilistic information is used in decisions made automatically (without human intervention) by computer programs. Thus, automated reasoning systems need to know how to reason with probabilistic information.

Despite the fact that quantitative logic programming has been studied intensely (cf. the works of van Emden, 1986; Shapiro, 1983; Fitting, 1988a, b; Blair and Subrahmanian, 1987, 1988; Kifer and Li, 1988; Kifer and Subrahmanian, 1992; and Morishita, 1989), no probabilistic foundation for multivalued logic programming has been developed thus far. There is no doubt that probability theory is the most widely accepted formalism for reasoning about chance and uncertainty. As logic programs are a natural formalism for designing rule based expert systems, it is of vital importance that they have the ability to reason with probabilistic information. The main aim of this paper is to propose and semantically characterize such a logic programming language.

In brief, the principal contributions of this paper are:

1. To design a logical framework within which probabilistic information can be easily expressed. This is done by extending the annotated logics introduced by Blair and Subrahmanian (1987) and Subrahmanian (1987, 1992) (i) to allow conjunctions and disjunctions to be annotated and (ii) to allow annotations to be closed intervals of truth values.

2. To study the semantics of this language, and to clearly understand the relationships between probability theory, model theory, fixpoint theory, and proof theory for such languages.

3. In particular, to show that the model-theoretic framework developed here satisfies the criteria proposed by Fenstad (1980) for a function to be called probabilistic.

4. Some complications that arise in (2) above are that even sets of definite-clause-like formulas may be inconsistent in a probabilistic sense. For instance, the probabilistic statement "The probability of event E lies in the range $[0.2, 0.3]$" is inconsistent with the probabilistic statement "The probability of E lies in the range $[0.5, 0.6]$." Our model theory appropriately handles such probabilistic phenomena.

5. To develop a query processing procedure for handling queries to such programs. The procedure is complicated by the fact that unification of conjunctions and disjunctions of atoms does not proceed in the classical way, and that mgu's may not be unique.

2. SYNTAX

Let L be a fixed first order language containing infinitely many variable symbols, and finitely many constant and predicate symbols, but no function symbols,[1] and let B_L be the Herbrand base of L. Thus, B_L is always finite.

DEFINITION 1. (i) $conj(B_L) = \{A_1 \wedge \cdots \wedge A_n \mid n \geq 1$ is an integer, $A_1, ..., A_n \in B_L$, and $\forall 1 \leq i, j \leq n, i \neq j \Rightarrow A_i \neq A_j\}$.

(ii) $disj(B_L) = \{A_1 \vee \cdots \vee A_n \mid n \geq 1$ is an integer, $A_1, ..., A_n \in B_L$, and $\forall 1 \leq i, j \leq n, i \neq j \Rightarrow A_i \neq A_j\}$.

Thus, $conj(B_L)$ and $disj(B_L)$ denote, respectively, the sets of all ground conjunctions and disjunctions formed by using *distinct* atoms in B_L. A conjunction or disjunction with repeated atoms is considered equivalent to one without repeated atoms, by simply deleting the repetitions.

DEFINITION 2. If A is an atom (not necessarily ground) and $\mu = [\alpha, \beta] \subseteq [0, 1]$, then $A : \mu$ is called a *p-annotated atom*. μ is called the *p-annotation* of A.[2] Similarly, if C is a conjunction and D is a disjunction (both of which are not necessarily ground), then $C : \mu$ and $D : \mu$ are called a *p-annotated conjunction* and a *p-annotated disjunction* respectively.

DEFINITION 3. A *basic formula* (not necessarily ground) is either a conjunction or a disjunction of atoms. Note that both disjunction and conjunction cannot occur simultaneously in one basic formula. Furthermore, let $bf(B_L)$ denote the set of all *ground* basic formulas using distinct atoms in B_L; i.e., $bf(B_L) = conj(B_L) \cup disj(B_L)$.

DEFINITION 4. If A is an atom, $F_1, ..., F_n$ are basic formulas, and $\mu, \mu_1, ..., \mu_n$ are p-annotations, then $A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ is called a *p-clause*. $A : \mu$ is called the *head* of this p-clause, while $F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ is called the *body*. We assume, usually, that $\mu, \mu_1, ..., \mu_n$ are all distinct from $[0, 1]$.

Intuitively, if $\mu = [\alpha, \beta]$, then $F : \mu$ is to read as: "The probability of F lies in the interval $[\alpha, \beta]$". Thus, to say that event F cannot occur, we

[1] The technical reason that function symbols are not supported will be clarified later.

[2] Note that here atoms are annotated with sets of truth values rather than with truth values as in Blair and Subrahmanian (1987). While sets of truth values may be used to annotate atoms, the resulting semantics prescribed by Blair and Subrahmanian (1987) is non-probabilistic.

merely say $F : [0, 0]$ which is read as: "the probability of F lies in the interval $[0, 0]$" which is the same as saying "the probability of event F is zero."

DEFINITION 5. A *probabilistic logic program* (p-program for short) is a finite set of p-clauses.

We say that $A : \mu$ is unifiable with $B : \mu'$ via θ iff A and B are unifiable via some substitution θ. Note that we are not defining the result of the unification yet. Also note that p-programs are different from the annotated programs of Blair and Subrahmanian (1987) in two ways: first, conjunctions and disjunctions are allowed to be annotated, and second, the annotations are sets of truth values rather than individual truth values. Both these distinctions have a significant impact on the semantics of annotated logics. Further distinctions from Blair and Subrahmanian (1987) are discussed in Examples 6 and 14. Note that p-programs are implicitly allowed to contain negation in clause heads because $A : [0, 0]$, intuitively corresponds to the classical logic sentence "A is false." Semantics of logic programs with negations in clause heads were studied first by Subrahmanian (1987) and later by Blair and Subrahmanian (1987).

EXAMPLE 1. A long distance phone company, on receiving customers requests for connections, tries to find reliable paths within a network of relay centers. (Here assume that reliability is defined as the probability that a connection is error-free throughout the connected period.) The company supports two types of direct connection between relay centers. Suppose a statistical survey reveals the following performance figures for these two types of connections:

(i) Type A connection on its own has a reliability of $90\% \pm 5\%$.

(ii) Type B connection, on the other hand, is more reliable, providing a reliability of over 90% on its own.

(iii) Suppose X, Y, and Z are three centers. X and Z are connected by a Type A connection, while Z and Y are connected by a path of reliability at least 85%. Then the resulting path from X to Y suffers a drop in reliability to the 80% to 95% range.

(iv) As part of a path, type B again is more reliable. If X and Z are connected by a type B connection and Z and Y are connected by a path of reliability at least 75%, then the resulting path from X to Y has a reliability of at least 85%.

Now the company can use the following p-program to find the reliability of paths from one center to another (clauses 1–4 correspond to points (i) to (iv), respectively):

$path(X, Y) : [0.85, 0.95] \leftarrow a(X, Y) : [1, 1]$

$path(X, Y) : [0.9, 1] \leftarrow b(X, Y) : [1, 1]$

$path(X, Y) : [0.8, 0.95] \leftarrow a(X, Z) : [1, 1] \wedge path(Z, Y) : [0.85, 1]$

$path(X, Y) : [0.85, 1] \leftarrow b(X, Z) : [1, 1] \wedge path(Z, Y) : [0.75, 1]$.

3. Fixpoint Semantics

Definition 6. An *atomic function* is a mapping $f: B_L \rightarrow \mathscr{C}[0, 1]$, where $\mathscr{C}[0, 1]$ denotes the set of all closed sub-intervals of the unit interval $[0, 1]$.

Note that the empty interval, denoted by $\varnothing$, is a closed interval of the form $[\alpha, \beta]$ with $\beta < \alpha$. Intuitively an atomic function assigns a probability range to each ground atom. In the situation when the empty interval $\varnothing$ is assigned to an atomic function, inconsistency seems to exist. We formalize the notion of probabilistic consistency in the next section.

Recall that p-programs allow non-atomic basic formulas to appear in the body but not in the head of p-clauses. Thus we need a mechanism to assign probability ranges to non-atomic formulas. Suppose $\mathbf{P}(E)$ is used to denote the probability of event E. Kolmogorov (1956) and Hailperin (1984) have shown that given distinct events E_1 and E_2, we cannot precisely specify $\mathbf{P}(E_1 \wedge E_2)$ from $\mathbf{P}(E_1)$ and $\mathbf{P}(E_2)$. But we can characterize precisely the *range* within which the probability of $(E_1 \wedge E_2)$ must lie. As Frechet (1935) has shown, $\max\{0, \mathbf{P}(E_1) + \mathbf{P}(E_2) - 1\} \leqslant \mathbf{P}(E_1 \wedge E_2) \leqslant \min\{\mathbf{P}(E_1), \mathbf{P}(E_2)\}$ represents the tightest bounds for $\mathbf{P}(E_1 \wedge E_2)$. This result can be generalized as shown below. In the sequel, a *world* is simply a Herbrand interpretation as defined in Lloyd (1987). Given the two events, there are four *possible* worlds: first, world K_1, in which the events E_1 and E_2 both occur; second, world K_2, in which E_1 occurs, but E_2 does not occur; third, world K_3, in which E_2 occurs while E_1 does not occur; and last, world K_4, in which neither E_1 nor E_2 occur. Suppose $\mathbf{P}(E_1) \in [\alpha_1, \beta_1] \subseteq [0, 1]$ and $\mathbf{P}(E_2) \in [\alpha_2, \beta_2] \subseteq [0, 1]$. Furthermore, let k_i be the probability that world K_i is the *actual* world. This situation can be expressed via the following linear program $\mathbf{Q}$:

$$0 \leqslant \alpha_1 \leqslant k_1 + k_2 \leqslant \beta_1 \leqslant 1,$$

$$0 \leqslant \alpha_2 \leqslant k_1 + k_3 \leqslant \beta_2 \leqslant 1,$$

$$\sum_{j=1}^{4} k_j = 1,$$

and

$$\text{for} \quad j = 1, ..., 4, \qquad k_j \geqslant 0.$$

As event E_1 occurs in worlds K_1 and K_2 which are mutually incompatible worlds, the probability of E_1 occurring is $(k_1 + k_2)$. As $\mathbf{P}(E_1)$ is known to be in $[\alpha_1, \beta_1]$, this gives rise to the first inequality in the linear program **Q**. The second inequality arises similarly when we consider E_2 instead of E_1. The third inequality says that the four possible worlds encompass all possibilities. The fourth inequality simply asserts that probabilities are non-negative.

To find the range for $\mathbf{P}(E_1 \wedge E_2)$, we need to solve the linear program **Q** for the parameter k_1 that represents the probability of the world in which E_1 and E_2 are both true. However, in general, there is no unique solution. Thus, we need to solve **Q** to find the minimal and maximal values of k_1. Likewise, to find the range for $\mathbf{P}(E_1 \vee E_2)$, we solve for the minimal and maximal values of $(k_1 + k_2 + k_3)$. Hereafter we use the notation $\mathbf{min}_{\mathbf{Q}}\, E$ and $\mathbf{max}_{\mathbf{Q}}\, E$ to denote the minimization and maximization of expression E subject to the linear program **Q** described above.

THEOREM 1. *For the linear program* **Q**,

$$0 \leqslant \alpha_1 \leqslant k_1 + k_2 \leqslant \beta_1 \leqslant 1,$$

$$0 \leqslant \alpha_2 \leqslant k_1 + k_3 \leqslant \beta_2 \leqslant 1,$$

$$\sum_{j=1}^{4} k_j = 1,$$

and

$$\textit{for} \quad j = 1, ..., 4, \qquad k_j \geqslant 0,$$

(i) $\mathbf{min}_{\mathbf{Q}}\, k_1 = max\{0, \alpha_1 + \alpha_2 - 1\}$,

(ii) $\mathbf{max}_{\mathbf{Q}}\, k_1 = min\{\beta_1, \beta_2\}$

(iii) $\mathbf{min}_{\mathbf{Q}}\, (k_1 + k_2 + k_3) = max\{\alpha_1, \alpha_2\}$, *and*

(iv) $\mathbf{max}_{\mathbf{Q}}\, (k_1 + k_2 + k_3) = min\{1, \beta_1 + \beta_2\}$.

Proof. (i) *Claim.* $\mathbf{min}_{\mathbf{Q}}\, k_1 = max\{0, \alpha_1 + \alpha_2 - 1\}$.

Case 1. $\alpha_1 + \alpha_2 \leqslant 1$. First observe that $k_1 = 0$, $k_2 = \alpha_1$, $k_3 = \alpha_2$, $k_4 = (1 - \alpha_1 - \alpha_2)$ is a solution to the linear program. Second, any solution to the linear program satisfies the constraint $k_1 \geqslant 0$. Therefore, it follows that $\mathbf{min}_{\mathbf{Q}}\, k_1 = 0 = max\{0, \alpha_1 + \alpha_2 - 1\}$.

Case 2. $\alpha_1 + \alpha_2 > 1$. First observe that $k_1 = (\alpha_1 + \alpha_2 - 1)$, $k_2 = (1 - \alpha_2)$, $k_3 = (1 - \alpha_1)$, $k_4 = 0$ is a solution to the linear program. Suppose there exists a solution such that $k_1 < (\alpha_1 + \alpha_2 - 1)$. That is, $k_1 = (\alpha_1 + \alpha_2 - 1 - \delta)$, for some $\delta > 0$. Then to satisfy the first inequality, $k_2 \geqslant (1 - \alpha_2 + \delta)$. Similarly to satisfy the second inequality, $k_3 \geqslant (1 - \alpha_1 + \delta)$. Therefore, it follows that $(k_1 + k_2 + k_3) \geqslant (\alpha_1 + \alpha_2 - 1 - \delta) + (1 - \alpha_2 + \delta) + (1 - \alpha_1 + \delta) = (1 + \delta) > 1$. Thus, $k_4 < 0$, which is a contradiction! Therefore, for all solutions to the linear program, $k_1 \geqslant (\alpha_1 + \alpha_2 - 1)$. Thus it follows that $\mathbf{min}_{\mathbf{Q}}\, k_1 = (\alpha_1 + \alpha_2 - 1) = max\{0, \alpha_1 + \alpha_2 - 1\}$. Combining the results for Case 1 and 2, claim (i) is proved.

(ii) *Claim.* $\mathbf{max}_{\mathbf{Q}}\, k_1 = min\{\beta_1, \beta_2\}$.

Case 1. $\beta_1 \leqslant \beta_2$. First observe that $k_1 = \beta_1$, $k_2 = 0$, $k_3 = (\beta_2 - \beta_1)$, $k_4 = (1 - \beta_2)$ is a solution to the linear program. Second, for all solutions to the linear program and in particular to the first inequality, $k_1 \leqslant \beta_1$, since $k_2 \geqslant 0$. Therefore, $\mathbf{max}_{\mathbf{Q}}\, k_1 = \beta_1 = min\{\beta_1, \beta_2\}$.

Case 2. $\beta_1 > \beta_2$. The proof that $\mathbf{max}_{\mathbf{Q}}\, k_1 = \beta_2 = min\{\beta_1, \beta_2\}$ is similar to the above. Combining the results for Case 1 and 2, claim (ii) is proved.

(iii) *Claim.* $\mathbf{min}_{\mathbf{Q}}\, (k_1 + k_2 + k_3) = max\{\alpha_1, \alpha_2\}$. We subtract the first two inequalities of **Q** from 1 to have the following inequalities:

$$1 - \alpha_1 \geqslant 1 - k_1 - k_2 = k_4 + k_3 \geqslant 1 - \beta_1,$$

$$1 - \alpha_2 \geqslant 1 - k_1 - k_3 = k_4 + k_2 \geqslant 1 - \beta_2.$$

It is then easy to see that k_4 plays the role of k_1 in **Q**. Hence, it follows from (ii) that $\mathbf{max}_{\mathbf{Q}}\, k_4 = min\{1 - \alpha_1, 1 - \alpha_2\}$. Thus, it is easy to check that, $\mathbf{min}_{\mathbf{Q}}\, (k_1 + k_2 + k_3) = 1 - \mathbf{max}_{\mathbf{Q}}\, k_4 = max\{\alpha_1, \alpha_2\}$.

(iv) *Claim.* $\mathbf{max}_{\mathbf{Q}}\, (k_1 + k_2 + k_3) = min\{1, \beta_1 + \beta_2\}$. The proof is similar to that of case (iii). This completes the proof of the theorem. ∎

We define two operators $\otimes$ and $\oplus$ that combine intervals according to Theorem 1.

DEFINITION 7. Let $[\alpha_1, \beta_1]$ and $[\alpha_2, \beta_2]$ be sub-intervals of $[0, 1]$. Define:

(1) the operator $\otimes$, where $[\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2] = [max\{0, \alpha_1 + \alpha_2 - 1\}, min\{\beta_1, \beta_2\}]$, and

(2) the operator $\oplus$, where $[\alpha_1, \beta_1] \oplus [\alpha_2, \beta_2] = [max\{\alpha_1, \alpha_2\}, min\{1, \beta_1 + \beta_2\}]$.

The following lemma shows a few properties of $\otimes$ and $\oplus$ that will be used in later proofs.

LEMMA 1. *Let* $[\alpha_1, \beta_1]$, $[\alpha_2, \beta_2]$, $[\alpha_3, \beta_3]$, $[\delta_1, \gamma_1]$, *and* $[\delta_2, \gamma_2]$ *all be sub-intervals of* $[0, 1]$.

(1) $\otimes$ *and* $\oplus$ *are commutative*; *e.g.*, $[\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2] = [\alpha_2, \beta_2] \otimes [\alpha_1, \beta_1]$.

(2) $\otimes$ *and* $\oplus$ *are associative*, *e.g.*, $([\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2]) \otimes [\alpha_3, \beta_3] = [\alpha_1, \beta_1] \otimes ([\alpha_2, \beta_2] \otimes [\alpha_3, \beta_3])$.

(3) $\otimes$ *and* $\oplus$ *are monotonic on both arguments*, *e.g.*, *if* $[\alpha_1, \beta_1] \supseteq [\delta_1, \gamma_1]$, *and* $[\alpha_2, \beta_2] \supseteq [\delta_2, \gamma_2]$, *then* $[\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2] \supseteq [\delta_1, \gamma_1] \otimes [\delta_2, \gamma_2]$.

(4) $\otimes$ *and* $\oplus$ *are strictly monotonic on both arguments*; *e.g.*, *replace the relation* $\supseteq$ *in case* 3 *above with* $\supset$.

(5) $\varnothing \oplus [\alpha_1, \beta_1] = \varnothing$ *and* $\varnothing \otimes [\alpha_1, \beta_1] = \varnothing$.

(6) $[\alpha_1, \beta_1] \oplus [0, 1] = [\alpha_1, 1]$ *and* $[\alpha_1, \beta_1] \otimes [0, 1] = [0, \beta_1]$.

Proof. (1) and (6) follow immediately from Definition 7. As for proofs of (2), (3), and (5), we only show the case for $\otimes$, as those for $\oplus$ are similar. The proof of (4) is a straightforward modification of the one for (3).

(i) *Claim.* $([\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2]) \otimes [\alpha_3, \beta_3] = [\alpha_1, \beta_1] \otimes ([\alpha_2, \beta_2] \otimes [\alpha_3, \beta_3])$. By Definition 7, $[\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2] = [max\{0, \alpha_1 + \alpha_2 - 1\}, min\{\beta_1, \beta_2\}]$. Then again by the same definition, $([\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2]) \otimes [\alpha_3, \beta_3] = [max\{0, max\{0, \alpha_1 + \alpha_2 - 1\} + \alpha_3 - 1\}, min\{\beta_3, min\{\beta_1, \beta_3\}\}]$. Since $\alpha_3 \leqslant 1$, $max\{0, max\{0, \alpha_1 + \alpha_2 - 1\} + \alpha_3 - 1\} = max\{0, \alpha_1 + \alpha_2 + \alpha_3 - 2\}$. In addition, $min\{\beta_3, min\{\beta_1, \beta_2\}\} = min\{\beta_1, \beta_2, u_3\}$. Therefore, $([\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2]) \otimes [\alpha_3, \beta_3] = [max\{0, \alpha_1 + \alpha_2 + \alpha_3 - 2\}, min \leqslant \beta_1, \beta_2, \beta_3\}]$. Similarly, $[\alpha_1, \beta_1] \otimes ([\alpha_2, \beta_2] \otimes [\alpha_3, \beta_3]) = [max\{0, max\{0, \alpha_2 + \alpha_3 - 1\} + \alpha_1 - 1\}, min\{\beta_1, min\{\beta_2, \beta_3\}\}]$. Since $\alpha_1 \leqslant 1$, $[\alpha_1, \beta_1] \otimes ([\alpha_2, \beta_2] \otimes [\alpha_3, \beta_3]) = [max\{0, \alpha_1 + \alpha_2 + \alpha_3 - 2\}, min\{\beta_1, \beta_2, \beta_3\}]$.

(ii) *Claim.* If $[\alpha_1, \beta_1] \supseteq [\delta_1, \gamma_1]$, and $[\alpha_2, \beta_2] \supseteq [\delta_2, \gamma_2]$, then $[\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2] \supseteq [\delta_1, \gamma_1] \otimes [\delta_2, \gamma_2]$. From Definition 7, $[\alpha_1, \beta_1] \otimes [\alpha_2, \beta_2] = [max\{0, \alpha_1 + \alpha_2 - 1\}, min\{\beta_1, \beta_2\}]$, and $[\delta_1, \gamma_1] \otimes [\delta_2, \gamma_2] = [max\{0, \delta_1 + \delta_2 - 1\}, min\{\gamma_1, \gamma_2\}]$. Therefore, given $\alpha_1 \leqslant \delta_1$, $\alpha_2 \leqslant \delta_2$, $\gamma_1 \leqslant \beta_1$, and $\gamma_2 \leqslant \beta_2$, it suffices to prove that (a) $max\{0, \alpha_1 + \alpha_2 - 1\} \leqslant max\{0, \delta_1 + \delta_2 - 1\}$, and (b) $min\{\gamma_1, \gamma_2\} \leqslant \{\beta_1, \beta_2\}$.

(a) *Case* 1. $\alpha_1 + \alpha_2 \leqslant 1$. Therefore, it follows that $max\{0, \delta_1 + \delta_2 - 1\} \geqslant 0 = max\{0, \alpha_1 + \alpha_2 - 1\}$.

Case 2. $\alpha_1 + \alpha_2 > 1$. Since $\alpha_1 \leqslant \delta_1$ and $\alpha_2 \leqslant \delta_2$, it follows that $(\delta_1 + \delta_2 - 1) \geqslant (\alpha_1 + \alpha_2 - 1) > 0$. Therefore, $max\{0, \delta_1 + \delta_2 - 1\} = (\delta_1 + \delta_2 - 1) \geqslant (\alpha_1 + \alpha_2 - 1) = max\{0, \alpha_1 + \alpha_2 - 1\}$. This completes the proof for (a).

(b) It is true that $min\{\gamma_1, \gamma_2\} \leqslant \gamma_1 \leqslant \beta_1$, and similarly $min\{\gamma_1, \gamma_2\} \leqslant \gamma_2 \leqslant \beta_2$. Therefore, it follows that $min\{\gamma_1, \gamma_2\} \leqslant min\{\beta_1, \beta_2\}$. This completes the proof for (b) and claim (ii).

(iii) *Claim.* $\varnothing \otimes [\alpha_1, \beta_1] = \varnothing$. This result can be readily seen from the proof of Theorem 1, as the constraint for $\varnothing$ is not satisfiable. ∎

Recall that p-programs allow non-atomic basic formulas to appear in the body but not in the head of p-clauses. As formalized in the following definition, a *formula function* determines assignments of probability ranges to non-atomic formulas by applying the operators $\otimes$ and $\oplus$ on the probability ranges assigned to atomic formulas.

DEFINITION 8. Given an atomic function $f: B_L \to \mathscr{C}[0, 1]$, a corresponding *formula function* $h: bf(B_L) \to \mathscr{C}[0, 1]$ is defined inductively as follows:

(i) $h(F) = f(F)$, if F is an atom,

(ii) $h(F_1 \wedge F_2) = h(F_1) \otimes h(F_2)$, where $(F_1 \wedge F_2)$ is in $bf(B_L)$, and

(iii) $h(F_1 \vee F_2) = h(F_1) \oplus h(F_2)$, where $(F_1 \vee F_2)$ is in $bf(B_L)$.

Thus, there is a correspondence between formula functions and atomic functions in the sense that given any formula function h, we can get an atomic function by restricting h to ground atoms. Likewise, given any atomic function f, Definition 8 allows us to obtain a formula function h. We now define an ordering on the set of formula functions.

DEFINITION 9. Given two formula functions h_1 and h_2, we say that $h_1 \leqslant h_2$ iff $\forall F \in bf(B_L)$, $h_1(F) \supseteq h_2(F)$.

DEFINITION 10. Let $\mathscr{FF}$ denote the set of all formula functions.

Note 1. Observe that arbitrary unions of closed intervals are not necessarily closed. To see this, suppose that our language consists of the single propositional symbol p, and that f_j, for $j \geqslant 1$, assigns the closed interval $[0, 1 - 1/2^j]$ to p. Then $\bigcup_{j \geqslant 1} f_j(p)$ is equal to the right-open interval $[0, 1)$. Thus, we cannot define greatest lower bounds by simply taking unions of closed sets. Instead, we take the topological *closure* (w.r.t. the order topology on the real line) of $[0, 1)$, which, as we expect, is the closed interval $[0, 1]$. (Recall that in any topological space X, the closure,

closure(Y), of a set $Y \subset X$ is the smallest closed set that contains Y. Closures are well defined in all topologies (Kelley, 1955).)

The following lemma tells us that $\mathcal{FF}$ forms a complete lattice.

LEMMA 2. *$\mathcal{FF}$ forms a complete lattice w.r.t. the ordering $\leqslant$ defined above.*

Proof. For every subset G of $\mathcal{FF}$, $\forall F \in bf(B_L)$, $\bigsqcup(G)(F) = \bigcap \{\mu \mid g(F) = \mu$ and $g \in G\}$ and $\bigsqcap(G)(F) = closure(\bigcup \{\mu \mid g(F) = \mu$ and $g \in G\})$. ∎

The top element of $\mathcal{FF}$ is the function h such that $\forall F \in bf(B_L)$, $h(F) = \varnothing$. The bottom element is the function h such that $\forall F \in bf(B_L)$, $h(F) = [0, 1]$. We now define a fixpoint operator T_P for program P.

DEFINITION 11. Let P be a p-program. $T_P: \mathcal{FF} \to \mathcal{FF}$ is defined inductively as follows:

(i) For all atoms $A \in B_L$, $T_P(h)(A) = \bigcap M_A$, where $M_A = \{\mu \mid A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ is a ground instance of a clause in P and $\forall i, 1 \leqslant i \leqslant n$, $h(F_i) \subseteq \mu_i\}$. If the set M_A is empty, then $T_P(h)(A) = [0, 1]$.

(ii) $T_P(h)(C_1 \wedge C_2) = T_P(h)(C_1) \otimes T_P(h)(C_2)$, where $(C_1 \wedge C_2)$ is in $bf(B_L)$, and

(iii) $T_P(h)(D_1 \vee D_2) = T_P(h)(D_1) \oplus T_P(h)(D_2)$, where $(D_1 \vee D_2)$ is in $bf(B_L)$.

In the following we prove that T_P is monotonic.

DEFINITION 12. Let F be a ground basic formula. Then let *rank*(F) be the number of distinct atoms occurring in F.

THEOREM 2. *T_P is monotonic. (That is, whenever $h_1 \leqslant h_2$, $T_P(h_1) \leqslant T_P(h_2)$.)*

Proof. Given a basic formula F, proceed by induction on *rank*(F) and apply part 3 of Lemma 1. ∎

DEFINITION 13. The *upward iteration* of T_P is defined as follows:

(i) $T_P \uparrow 0 = \perp$; i.e., $T_P \uparrow 0$ is the function that assigns $[0, 1]$ to all $F \in bf(B_L)$ and

(ii) $T_P \uparrow \alpha = T_P(T_P \uparrow (\alpha - 1))$, where α is a successor ordinal whose immediate predecessor is denoted by $(\alpha - 1)$ and

(iii) $T_P \uparrow \lambda = \bigsqcup \{T_P \uparrow \alpha \mid < \lambda\}$, where λ is a limit ordinal.

EXAMPLE 2. T_P may not always be continuous. Let $q : [0, 0.5] \leftarrow p : [0, 0]$ be the only clause of a probabilistic program P. Consider the following directed subset of $\mathcal{FF}$: $G = (g_j)_{j=0}^{\infty}$, where $g_j(p) = [0, 1/2^j]$, $g_j(q) = [0, 1]$. Then, $(\bigsqcup G)(p) = [0, 0]$, and $T_P(\bigsqcup G)(q) = [0, 0.5]$. However, for all j, $T_P(g_j)(q) = [0, 1]$. Therefore, it follows that $\bigsqcup_j (T_P(g_j)(q)) = [0, 1] \neq T_P(\bigsqcup G)(q)$.

To prove Lemma 4, we prove the following lemma first.

LEMMA 3. *For all* $F \in bf(B_L)$, $T_P \uparrow \omega(F) = \mu \Rightarrow \exists n < \omega$ *such that* $T_P \uparrow n(F) = \mu$.

Proof. Proceed by induction on *rank*(F).

Base Case: *rank*$(F) = 1$. Then $F \equiv A$ for some atom A. Suppose $T_P \uparrow \omega(A) \neq T_P \uparrow n(A)$ for all $n < \omega$. Since $T_P \uparrow 0(A) \supseteq T_P \uparrow 1(A) \supseteq \cdots \supseteq T_P \uparrow \omega(A)$, there exists an ascending sequence of integers $\alpha_0, \alpha_1, \ldots$ suchthat $T_P \uparrow \alpha_0(A) \supset T_P \uparrow \alpha_1(A) \supset \cdots$. In particular, since $T_P \uparrow \omega(A) = \bigsqcup \{T_P \uparrow n(A) \mid n < \omega\}$, and $T_P \uparrow \omega(A) \neq T_P \uparrow n(A)$ for all $n < \omega$, the sequence $\alpha_0, \alpha_1, \ldots$ must be infinite. But for each α_j in the sequence, $T_P \uparrow \alpha_j(A) = \bigsqcup X_j$, where $X_j \subseteq \mathcal{A} = \{\mu \mid \mu$ is the annotation of the head of a clause which unifies with $A\}$. Therefore, there exists a corresponding infinite sequence $X_0, X_1, \ldots$ of subsets of $\mathcal{A}$ such that $X_0 \subset X_1 \subset \cdots$. However, since program P consists of only a finite set of clauses. $\mathcal{A}$ and therefore the number of subsets of $\mathcal{A}$ must both be finite. Therefore, there exists $i < j$ such that $X_i = X_j$, a contradiction!

Inductive Case: *rank*$(F) > 1$. Then F is either a conjunction or a disjunction.

Case 1. $F \equiv C_1 \wedge C_2$. By Definition 11, $T_P \uparrow \omega(F) = T_P \uparrow \omega(C_1 \wedge C_2) = T_P \uparrow \omega(C_1) \otimes T_P \uparrow \omega(C_2)$. But by the induction hypothesis, there exist $n_1 < \omega$ such that $T_P \uparrow n_1(C_1) = T_P \uparrow \omega(C_1)$, and $n_2 < \omega$ such that $T_P \uparrow n_2(C_2) = T_P \uparrow \omega(C_2)$. Pick $n = max\{n_1, n_2\} < \omega$. Then it follows that $T_P \uparrow n(F) = T_P \uparrow n(C_1) \otimes T_P \uparrow n(C_2) = T_P \uparrow \omega(C_1) \otimes T_P \uparrow \omega(C_2) = T_P \uparrow \omega(F)$.

Case 2. $F \equiv D_1 \vee D_2$. The proof is similar to the one for conjunctions in Case 1. This completes the induction and the proof of the lemma. ∎

Despite the fact that T_P is not always continuous, the following result holds.

LEMMA 4. *There exists an integer* $n < \omega$ *such that* $T_P \uparrow n = ifp(T_P)$.

Proof. Immediate consequence of the lemma above and the fact that $bf(B_L)$ is finite. ∎

Lemma 4 tells us that the T_P operator always achieves a fixpoint after a finite iteration.

4. Probabilistic Model Theory

In this section, we present a model theory that captures the uncertainty described in p-programs. We introduce notions such as probabilistic truth values of formulas and probabilistic interpretations and models for p-programs. We also study the relationships between formula functions and probabilistic interpretations, and between fixpoints of T_P and probabilistic models of P.

Definition 14. Consider any enumeration of 2^{B_L}, i.e., $2^{B_L} = \{K_1, ..., K_r\}$ for some integer r. A *probabilistic kernel interpretation* is a mapping KI: $2^{B_L} \to [0, 1]$ such that for all $K_j \in 2^{B_L}$, $KI(K_j) \geq 0$ and $\sum_{K_j \in 2^{B_L}} KI(K_j) = 1$. Hereafter, we denote $KI(K_j)$ by k_j.

Intuitively, probabilistic kernel interpretations assume that the "real" world is definite, i.e., there is some set of propositions that are true, and some set of propositions that are false. However, it is not sure which of the various "possible worlds" (i.e., worlds are just 2-valued interpretations) is the right one. Hence, a kernel interpretation assigns a probability to each 2-valued interpretation of our language. As 2^{B_L} consists of all possible worlds, the sum of all probabilities assigned must be 1. Any two distinct worlds are mutually incompatible as they must differ on at least one atom. Hence, we can compute the probability of a formula F in a kernel interpretation KI by just summing up the probabilities assigned to those worlds in 2^{B_L} in which F is true.

Note on Notation. Throughout this paper, given a kernel interpretation KI, we use $K_1, ..., K_n$ to denote the elements of 2^{B_L} and $k_1, ..., k_n$ to denote $KI(K_1), ..., KI(K_n)$ respectively.

Example 3. Suppose L consists of two propositional symbols p and q. Then $KI(\varnothing) = 0.4$, $KI(\{p\}) = 0.25$, $KI(\{q\}) = 0.35$, $KI(\{p, q\}) = 0$ is a probabilistic kernel interpretation. Intuitively, KI says that the probability that both p and q are false is 0.4, the probability that p is true in the real world but q is false is 0.25, the probability that q is true in the real world but p is false is 0.35, and the probability that both p and q are true in the real world is 0.

Recall from the previous section that a formula function h specifies a probability range for each basic formula. Given h, we would like to find

those kernel interpretations whose probability assignments to basic formulas fall within the ranges specified by h. The notion defined below captures this idea. In the sequel, we will assume we are speaking about some fixed language L with $2^{B_L} = \{K_1, ..., K_r\}$ and some fixed probabilistic kernel interpretation KI such that $KI(K_j) = k_j$ for all $1 \leqslant j \leqslant r$.

DEFINITION 15. (1) Let h be a formula function, and let $\mathscr{LP}(h)$ denote the linear program

$$\forall F_i \in bf(B_L), \alpha_i \leqslant \left(\sum_{K_j \models F_i \text{ and } K_j \in 2^{B_L}} k_j \right) \leqslant \beta_i, \qquad \text{where} \quad h(F_i) = [\alpha_i, \beta_i];$$

$$\sum_{K_j \in 2^{B_L}} k_j = 1;$$

and

$$\forall K_j \in 2^{B_L}, k_j \geqslant 0.$$

Let $\mathscr{KI}(h)$ be the solution set of the above linear program.

(2) Let f be an atomic function. Then let $\mathscr{LP}(f)$ denote a linear program similar to the one above, except that there is one constraint based on the range assigned by f to each ground atom, instead of for each ground basic formula. Similarly, let $\mathscr{KI}(f)$ be the solution set of $\mathscr{LP}(f)$.

Note that in the above linear program, the K_j's are 2-valued interpretations and the F_i's are basic formulas. All basic formulas are formulas of classical logic. Hence, the expression $K_j \models F$ is to be read: "K_j satisfies F" where satisfaction is defined as in classical logic (Boolos and Jeffrey, 1980; Shoenfield, 1967). Note that every solution to $\mathscr{LP}(h)$ is a kernel interpretation that satisfies h. Hence, we use the notation $\mathscr{KI}(h)$ to denote the family of kernel interpretations that satisfy h.

Also note that each basic formula in our language generates an inequality in the above linear program. If function symbols are allowed in our language, then $bf(B_L)$ will be infinite and the linear program will have infinitely many constraints and k_j's. As far as we know, current technology on the theory of linear programming in infinite dimensional space (Anderson and Nash, 1987) can only deal with *semi-infinite* linear programs which have either infinitely many variables (k_j's in our case) or infinitely many constraints, but not both. This is the technical reason that function symbols are disallowed in our framework. In the remainder of this section, we study the relationship between a kernel interpretation and a probabilistic interpretation, and more importantly the relationship between fixpoints and probabilistic models. While we discuss all these notions and

relationships in detail very shortly, we first examine when a family of kernel interpretations is non-empty.

Consider the atomic function f associated, per Definition 8, with a given formula function h. As defined in Definition 8, for all $A \in B_L$, $f(A) = h(A)$. In the following, we show that the family of kernel interpretations for a formula function h is the same as the family for the atomic function f associated with h; that is, $\mathscr{KI}(h) = \mathscr{KI}(f)$. The reason is that the constraints in $\mathscr{LP}(h)$ for non-atomic formulas are redundant and can therefore be discarded without altering the solution set to the linear program. These constraints are redundant because, as we recall from the previous section, probability ranges are propagated to non-atomic formulas through repeated applications of $\otimes$ and $\oplus$ on the ranges assigned to atoms. Hence, in finding solutions to the entire set of constraints, it suffices to consider constraints for the atoms only. In particular, Lemma 5 below states that in the presence of the constraints for $C_1, C_2 \in conj(B_L)$, the constraint for $C_1 \wedge C_2$ can be ignored.

LEMMA 5. *Let Ω be a subset of $\mathscr{LP}(h)$ for some formula function h (i.e., a set of linear constraints defining a linear program). Further suppose that Ω contains at least the following constraints*:

(i) $R \equiv \alpha_1 \leqslant (\sum_{K_j \models C_1 \text{ and } K_j \in 2^{B_L}} k_j) \leqslant \beta_1$ *(i.e., the constraint for $h(C_1)$ for some $C_1 \in conj(B_L)$)*,

(ii) $S \equiv \alpha_2 \leqslant (\sum_{K_j \models C_2 \text{ and } K_j \in 2^{B_L}} k_j) \leqslant \beta_2$ *(i.e., the constraint for $h(C_2)$ for some $C_2 \in conj(B_L)$)*,

(iii) $T \equiv max\{0, \alpha_1 + \alpha_2 - 1\} \leqslant (\sum_{K_j \models C_1 \wedge C_2 \text{ and } K_j \in 2^{B_L}} k_j) \leqslant min\{\beta_1, \beta_2\}$ *(i.e., the constraint for $h(C_1 \wedge C_2)$)*,

(iv) $I \equiv \sum_{K_j \in 2^{B_L}} k_j = 1$, *and*

(v) $M \equiv \forall K_j \in 2^{B_L}, k_j \geqslant 0$.

Then the solution set for $(\Omega - \{T\})$ is the same as that for Ω.

Proof. (i) *Claim.* The solution set for Ω is contained in the solution set for $(\Omega - \{T\})$. For every solution $\bar{k}$ in the solution set for Ω, it is obvious that $\bar{k}$ is also a solution to $(\Omega - \{T\})$.

(ii) *Claim.* The solution set for $(\Omega - \{T\})$ is contained in the solution set for Ω. Let $\bar{k}$ be a solution to $(\Omega - \{T\})$. Let $i_1 = (\sum_{K_j \models C_1 \wedge C_2 \text{ and } K_j \in 2^{B_L}} k_j)$, $i_2 = (\sum_{K_j \models C_1 \wedge \neg C_2 \text{ and } K_j \in 2^{B_L}} k_j)$, $i_3 = (\sum_{K_j \models \neg C_1 \wedge C_2 \text{ and } K_j \in 2^{B_L}} k_j)$, and $i_4 = (\sum_{K_j \models \neg C_1 \wedge \neg C_2 \text{ and } K_j \in 2^{B_L}} k_j)$. Then constraints R, S, T and I can be rewritten as:

(i) $R \equiv \alpha_1 \leqslant (i_1 + i_2) \leqslant \beta_1$,

(ii) $S \equiv \alpha_2 \leqslant (i_1 + i_3) \leqslant \beta_2$,

(iii) $T \equiv max\{0, \alpha_1 + \alpha_2 - 1\} \leqslant i_1 \leqslant min\{\beta_1, \beta_2\}$, and

(iv) $I \equiv \sum_{j=1}^{4} i_j = 1$.

Now let N be the constraint $N \equiv i_j \geqslant 0$ for $j = 1, ..., 4$. Note that whenever constraint M is satisfied, constraint N is satisfied. Since constraint $M \in \Omega$, $\bar{k}$ is also a solution to $(\Omega - \{T\} \cup \{N\})$. Now rewrite $(\Omega - \{T\} \cup \{N\}) \equiv (\{R, S, I, N\}) \cup \Omega_1$ for some set Ω_1 of constraints. But recall that the set $\{R, S, I, N\}$ is the same linear program used in Theorem 1. Then by Theorem 1, $\mathbf{min}_{\mathbf{Q}}\, i_1$ and $\mathbf{max}_{\mathbf{Q}}\, i_1$ are $max\{0, \alpha_1 + \alpha_2 - 1\}$ and $min\{\beta_1, \beta_2\}$, respectively; i.e., $max\{0, \alpha_1 + \alpha_2 - 1\} \leqslant i_1 \leqslant min\{\beta_1, \beta_2\}$. In other words, every solution of $\{R, S, I, N\}$ satisfies constraint T automatically, and therefore every solution of $\{R, S, I, N\}$ is a solution of $\{R, S, I, N, T\}$. Since $\bar{k}$ is a solution to $(\Omega - \{T\} \cup \{N\})$, $\bar{k}$ satisfies both Ω_1 and $\{R, S, I, N\}$. Thus, $\bar{k}$ satisfies both Ω_1 and $\{R, S, I, N, T\}$. But $\Omega_1 \cup (\{R, S, I, N, T\}) \equiv (\Omega - \{T\} \cup \{N\}) \cup \{T\} \equiv \Omega \cup \{N\}$. Therefore, k is also a solution to $(\Omega \cup \{N\})$. However as argued above, since constraint $M \in \Omega$, it follows that the solution set for $(\Omega \cup \{N\})$ is the same as that for Ω. Therefore, $\bar{k}$ is in the solution set for Ω. This completes the proof of the lemma. ∎

A similar lemma exists for disjunctions; that is, given the constraints for $D_1, D_2 \in disj(B_L)$, the constraint for $D_1 \vee D_2$ can be discarded. The proof can be easily obtained by modifying the above proof to consider the redundant constraint T to be

$$T \equiv max\{\alpha_1, \alpha_2\} \leqslant \left(\sum_{K_j \models D_1 \vee D_2 \text{ and } K_j \in 2^{B_L}} k_j \right) \leqslant min\{1, \beta_1 + \beta_2\},$$

where α_1, α_2, β_1, and β_2 are the lower bounds and upper bounds of the constraints for D_1 and D_2, respectively. By repeated applications of the above lemmas, we show in the following theorem that all constraints for non-atomic formulas are redundant and can therefore be ignored in considering the family of kernel interpretations that satisfy a formula function.

THEOREM 3. *Given a formula function h, $\mathscr{KI}(h) = \mathscr{KI}(f)$, where f is the atomic function associated with h.*

Proof. Consider the linear program $\mathscr{LP}(h)$. Pick any conjunction $C_1 \wedge C_2 \in conj(B_L)$ of the maximal rank. This is always possible as B_L is finite (though there may be several different choices for picking $C_1 \wedge C_2$). Let the constraint for $C_1 \wedge C_2$ in $\mathscr{LP}(h)$ be T, the one for C_1 be R, and the one for C_2 be S. Since $\mathscr{LP}(h)$ contains constraints I and M as defined in Lemma 5, the lemma guarantees that the solution set for

$(\mathscr{LP}(h) - \{T\})$ is the same as the one for $\mathscr{LP}(h)$, i.e., $\mathscr{KI}(h)$. Note that $(\mathscr{LP}(h) - \{T\})$ still contains constraints I and M. Thus among those conjunctions with *rank* $\geqslant 2$ and whose constraints have not been deleted, the lemma can be applied repeatedly to a conjunction of maximal rank. This iterative process stops when the remaining linear program only contains constraints for atoms, disjunctions, and constraints I and M—which is denoted by $(\mathscr{LP}(h) - \Omega_1)$ for some set Ω_1 of constraints. However, each application of the lemma guarantees that the solution sets before and after a deletion are identical. Therefore, $\mathscr{KI}(h)$ is identical to the solution set of $(\mathscr{LP}(h) - \Omega_1)$. Now constraints for disjunctions with *rank* $\geqslant 2$ are similarly deleted, based on repeated applications of the lemma for disjunctions. This iterative process stops when the remaining linear program only contains constraints for atoms and constraints I and M—which becomes $\mathscr{LP}(f)$! However, each application of the lemma guarantees that the solution sets before and after a deletion are identical. Therefore, $\mathscr{KI}(f)$, the solution set of $\mathscr{LP}(f)$, is identical to the solution set of $(\mathscr{LP}(h) - \Omega_1)$ which in turn is identical to $\mathscr{KI}(h)$. ▌

Theorem 3 demonstrates that from now on, we need to consider only linear programs associated with atomic functions rather than those associated with formula functions.

EXAMPLE 4. Suppose $B_L = \{A, B, C\}$. Then there are eight different Herbrand Interpretations K_1 to K_8, as summarized by the truth table below in the usual way:

	A	B	C
K_1	1	1	1
K_2	1	1	0
K_3	1	0	1
K_4	1	0	0
K_5	0	1	1
K_6	0	1	0
K_7	0	0	1
K_8	0	0	0

Thus, K_1 represents the Herbrand Interpretation containing A, B, and C, and so on. Suppose for some formula function h, $h(A) = [\alpha_1, \beta_1]$, $h(B) = [\alpha_2, \beta_2]$, and $h(C) = [\alpha_3, \beta_3]$. Then since A is true in the classical 2-valued sense in K_1, K_2, K_3 and K_4, the constraint for A is

$$\alpha_1 \leqslant k_1 + k_2 + k_3 + k_4 \leqslant \beta_1.$$

(As usual, k_i denotes the probability assigned to the Herbrand interpretation K_i.) Similarly, the constraints for B and C are as follows:

$$\alpha_2 \leqslant k_1 + k_2 + k_5 + k_6 \leqslant \beta_2,$$
$$\alpha_3 \leqslant k_1 + k_3 + k_5 + k_7 \leqslant \beta_3.$$

For the given Herbrand Base B_L, the set of all basic formulas $bf(B_L) = \{A,\ B,\ C,\ A \wedge B,\ A \wedge C,\ B \wedge C,\ A \wedge B \wedge C,\ A \vee B,\ A \vee C,\ B \vee C,\ A \vee B \vee C\}$. Then according to Definition 8, the ranges for conjunctions are computed using the operator $\otimes$. Hence, since $A \wedge B$ is true in K_1 and K_2, the constraint for $A \wedge B$ is

$$max\{0, \alpha_1 + \alpha_2 - 1\} \leqslant k_1 + k_2 \leqslant min\{\beta_1, \beta_2\}.$$

Similarly, the constraints for $A \wedge C$, $B \wedge C$, and $A \wedge B \wedge C$ are respectively

$$max\{0, \alpha_1 + \alpha_3 - 1\} \leqslant k_1 + k_3 \leqslant min\{\beta_1, \beta_3\},$$
$$max\{0, \alpha_2 + \alpha_3 - 1\} \leqslant k_1 + k_5 \leqslant min\{\beta_2, \beta_3\},$$

and

$$max\{0, \alpha_1 + \alpha_2 + \alpha_3 - 2\} \leqslant k_1 \leqslant min\{\beta_1, \beta_2, \beta_3\}.$$

As for disjunctions, again by Definition 8, the ranges are computed using the operator $\oplus$. It can be easily verified that the constraints for $A \vee B$, $A \vee C$, $B \vee C$ and $A \vee B \vee C$ are respectively:

$$max\{\alpha_1, \alpha_2\} \leqslant k_1 + \cdots + k_6 \leqslant min\{1, \beta_1 + \beta_2\},$$
$$max\{\alpha_1, \alpha_3\} \leqslant k_1 + k_2 + k_3 + k_4 + k_5 + k_7 \leqslant min\{1, \beta_1 + \beta_3\},$$
$$max\{\alpha_2, \alpha_3\} \leqslant k_1 + k_2 + k_3 + k_5 + k_6 + k_7 \leqslant min\{1, \beta_2 + \beta_3\},$$

and

$$max\{\alpha_1, \alpha_2, \alpha_3\} \leqslant k_1 + \cdots + k_7 \leqslant min\{1, \beta_1 + \beta_2 + \beta_3\}.$$

Now by applying the constraint deletion process described in the proof of Theorem 3, the constraint for $A \wedge B \wedge C$ is deleted first, followed by the constraints for $A \wedge B$, $A \wedge C$, $B \wedge C$, $A \vee B \vee C$, $A \vee B$, $A \vee C$, and $B \vee C$. What remains is the linear program

$$\alpha_1 \leqslant k_1 + k_2 + k_3 + k_4 \leqslant \beta_1,$$
$$\alpha_2 \leqslant k_1 + k_2 + k_5 + k_6 \leqslant \beta_2,$$
$$\alpha_3 \leqslant k_1 + k_3 + k_5 + k_7 \leqslant \beta_3,$$
$$k_1 + \cdots + k_8 = 1,$$

and

$$k_1, ..., k_8 \geq 0,$$

which according to Theorem 3, has the same solution set as the original $\mathscr{LP}(h)$.

In the following, we characterize those formula functions h (atomic functions) whose family $\mathscr{KI}(h)$ of kernel interpretations is non-empty.

DEFINITION 16. A formula function h is *fully defined* iff for all $F \in bf(B_L)$, $\varnothing \subset h(F) \subseteq [0, 1]$. An atomic function f is fully defined iff for all $A \in B_L$, $\varnothing \subset f(A) \subseteq [0, 1]$.

Intuitively, the assignment of the empty interval to an atom (or basic formula) tells us that there is no way of assigning a probability to that atom or formula. Thus, there seems to be an inconsistency concerning that atom or formula. This is what the definition of "full definedness" tries to capture.

LEMMA 6. *Let f be an atomic function, and h be the formula function associated with f. Then h is fully defined iff f is fully defined.*

Proof. (i) *Claim.* If f is not fully defined, then h is not fully defined. There exists an atom A in B_L such that the condition $\varnothing \subset f(A) \subseteq [0, 1]$ is violated. But since $h(A) = f(A)$, it is not true that $\varnothing \subset h(A) \subseteq [0, 1]$. It follows that h is not fully defined.

(ii) *Claim.* If f is fully defined, then h is fully defined. Let F be any ground formula in $bf(B_L)$. Proceed by induction on $rank(F)$.

Base Case: $rank(F) = 1$. Then F is a ground atom. Since f is fully defined and for all $F \in B_L$, $h(F) = f(F)$, it follows immediately that $\varnothing \subset h(F) \subseteq [0, 1]$.

Inductive Case: $rank(F) > 1$. Then F is either a conjunction or a disjunction.

Case 1. $F \equiv C_1 \wedge C_2$. From Definition 8, $h(F) = h(C_1) \otimes h(C_2)$. From the induction hypothesis, $\varnothing \subset h(C_1)$ and $\varnothing \subset (C_2)$. Then it follows immediately from Lemma 1 that $\varnothing = \varnothing \otimes \varnothing \subset h(C_1) \otimes h(C_2)$. This completes the proof for case 1.

Case 2. $F \equiv D_1 \vee D_2$. The proof is similar to the one for conjunctions in case 1. This completes the induction and the proof of the lemma. ∎

We are now in a position to characterize formula functions whose families of kernel interpretations are non-empty. Theorem 4 tells us that if

a formula function h is fully defined, then the set $\mathscr{LP}(h)$ of inequalities is guaranteed to possess at least one solution.

THEOREM 4. *If a formula function h is fully defined, then $\mathscr{KI}(h)$ is non-empty.*

Proof. From Lemma 6 above, and Theorem 3, it suffices to prove that if f, the atomic function associated with h, is fully defined, then $\mathscr{KI}(f)$ is non-empty. Without loss of generality, consider an arbitrary enumeration $A_1, ..., A_n$ of B_L (where $|B_L| = n$) such that $f(A_i) = [\alpha_i, \beta_i]$ for $1 \leqslant i \leqslant n$, and such that $\alpha_1 \leqslant \alpha_2 \leqslant \cdots \leqslant \alpha_n$. Based on this enumeration, the following table represents an enumeration of 2^{B_L}:

	A_1	A_2	$\cdots$	A_n
K_1	1	1	$\cdots$	1
K_2	⋮	⋮		0
⋮	⋮	⋮		
K_{2^n-2}	1	1		0
K_{2^n-2+1}	1	0		1
⋮	⋮	⋮		
K_{2^n-1}	1	0		0
K_{2^n-1+1}	0	1		1
⋮	⋮	⋮		
$K_{2^n-1+2^n-2}$	0	1		0
$K_{2^n-1+2^n-2+1}$	0	0		1
⋮	⋮	⋮		
K_{2^n}	0	0		0

For example, K_1 represents the 2-valued interpretation (i.e., world) $\{A_1, ..., A_n\}$, K_2 the 2-valued interpretation $\{A_1, ..., A_{n-1}\}$, and so forth. Under this enumeration, the system of linear inequalities $\mathscr{LP}(f)$ defined in Definition 15 becomes

$$\alpha_1 \leqslant k_1 + k_2 + \cdots + k_{2^n-1} \leqslant \beta_1,$$

$$\alpha_2 \leqslant (k_1 + \cdots + k_{2^n-2}) + (k_{2^n-1+1} + \cdots + k_{2^n-1+2^n-2}) \leqslant \beta_2,$$

$$\vdots$$

$$\alpha_n \leqslant k_1 + k_3 + \cdots + k_{2^n-1} \leqslant \beta_n,$$

and

$$\sum_{j=1}^{2^n} k_j = 1.$$

Then construct a solution for $\mathscr{KI}(f)$ by considering each inequality in turn:

(i) Take the first inequality and set $k_1 = \alpha_1$, $k_2 = \cdots = k_{2^{n-1}} = 0$. Then, $k_1 + \cdots + k_{2^{n-1}} = \alpha_1$, and therefore the first inequality is satisfied, as f is fully defined and $\alpha_1 \leqslant \beta_1$. In addition, $k_1, ..., k_{2^{n-1}} \geqslant 0$.

(ii) Take the second inequality and set $k_{2^{n-1}+1} = \alpha_2 - \alpha_1$, $k_{2^{n-1}+2} = \cdots = k_{2^{n-1}+2^{n-2}} = 0$. Recall that $\alpha_1 \leqslant \alpha_2$ and hence $k_{2^{n-1}+1} \geqslant 0$. Then $\sum_{j=1}^{2^{n-2}} k_j + \sum_{j=2^{n-1}+1}^{2^{n-1}+2^{n-2}} k_j = k_1 + k_{2^{n-1}+1} = \alpha_1 + (\alpha_2 - \alpha_1) = \alpha_2$. Therefore, the second inequality is satisfied as f is fully defined and $\alpha_2 \leqslant \beta_2$. In addition, $k_1, ..., k_{2^{n-1}+2^{n-2}} \geqslant 0$, and $k_1 + \cdots + k_{2^{n-1}+2^{n-2}} = \alpha_2$.

(iii) Continue this process one inequality at a time. Finally, when the last inequality is considered, set $k_{2^n-1} = \alpha_n - \alpha_{n-1}$. Then $\sum_{j=1}^{2^{n-1}} k_{2j-1} = \alpha_n$, satisfying the last inequality as $\alpha_n \leqslant \beta_n$. In addition, $k_1, ..., k_{2^n-1} \geqslant 0$, and $k_1 + \cdots + k_{2^n-1} = \alpha_n$.

(iv) Finally, to satisfy the condition that all k_j's add up to 1, set $k_{2^n} = 1 - \alpha_n \geqslant 0$. Thus whenever f is fully defined, $\mathscr{KI}(f)$ is non-empty. ∎

Theorem 4 guarantees that the linear program generated by a fully defined formula function is always solvable. We show below an example to illustrate how the proof of Theorem 4 works.

EXAMPLE 5. Continue with the situation described in Example 4. Without loss of generality, suppose the enumeration A, B, and C corresponds to the one where $\alpha_1 \leqslant \alpha_2 \leqslant \alpha_3$. Then according to the construction shown in the above proof, $k_1 = \alpha_1$, $k_2 = k_3 = k_4 = 0$, $k_5 = \alpha_2 - \alpha_1$, $k_6 = 0$, $k_7 = \alpha_3 - \alpha_2$, $k_8 = 1 - \alpha_3$ is a solution to the linear program

$$\alpha_1 \leqslant k_1 + k_2 + k_3 + k_4 \leqslant \beta_1,$$
$$\alpha_2 \leqslant k_1 + k_2 + k_5 + k_6 \leqslant \beta_2,$$
$$\alpha_3 \leqslant k_1 + k_3 + k_5 + k_7 \leqslant \beta_3,$$
$$k_1 + \cdots + k_8 = 1,$$

and

$$k_1, ..., k_8 \geqslant 0.$$

DEFINITION 17. A probabilistic kernel interpretation KI can be extended to a *probabilistic interpretation* which is a mapping I from formulas to $[0, 1]$ in the following way: For all formulas F, $I(F) = (\sum_{K_j \models F \text{ and } K_j \in 2^{B_L}} k_j)$.

Theorem 4 is significant as it guarantees that fully defined functions can

always be extended to probabilistic interpretations that assign probabilistic truth values to formulas. The two lemmas below show that this way of assigning probabilistic truth values satisfies many general properties of probability. Hereafter whenever no confusion arises, we simply use I to denote a probabilistic interpretation, without referring to the probabilistic kernel interpretation KI from which I is generated.

LEMMA 7 (Hailperin). *Suppose that KI is a probabilistic kernel interpretation and that I is the probabilistic interpretation associated with KI. Then the following conditions hold:*

(i) $I(\phi) = 0$, *if* $\phi \Rightarrow A \wedge \neg A$, *for some* A,

(ii) $I(\phi) \leqslant I(\psi)$, *if* $\phi \Rightarrow \psi$,

(iii) $I(\neg\phi) = 1 - I(\phi)$, *and*

(iv) $I(\phi \vee \psi) = I(\phi) + I(\psi)$, *if* $\phi \wedge \psi \Rightarrow A \wedge \neg A$, *for some* A.

Fenstad (1980) has identified the following requirements for defining a probability function p on a first-order language L:

(i) $p(\phi \vee \psi) + p(\phi \wedge \psi) = p(\phi) + p(\psi)$,

(ii) $p(\neg\phi) = 1 - p(\phi)$,

(iii) $p(\phi) = p(\psi)$, if ϕ and ψ are logically equivalent in L, and

(iv) $p(\phi) = 1$, if ϕ is provable in L.

It is obvious from the above lemma that probabilistic interpretations satisfy the last three requirements of Fenstad. The following lemma shows that probabilistic interpretations also satisfy the first requirement. The proof is straightforward.

LEMMA 8. *Let ϕ and ψ be arbitrary formulas in language L. Suppose that KI is a probabilistic kernel interpretation and that I is the probabilistic interpretation associated with KI. Then $I(\phi \vee \psi) + I(\phi \wedge \psi) = I(\phi) + I(\psi)$.*

Recall that for every formula function h, there is a corresponding family of kernel interpretations $\mathscr{K}\mathscr{I}(h)$. And as defined above, for each of these kernel interpretations, there is a corresponding probabilistic interpretation I. Therefore, associated with h is a family of probabilistic interpretations, denoted by $\mathscr{I}(h)$.

DEFINITION 18. Suppose KI is a probabilistic kernel interpretation, and I is the probabilistic interpretation associated with KI. Also let F be a basic formula and $F_0, \ldots, F_n \in bf(B_L)$, and $\mu, \mu_0, \ldots, \mu_n \subseteq [0, 1]$.

(i) $I \models F_1 : \mu$ iff $I(F_1) \in \mu$,

(ii) $I \models (F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ iff for all $1 \leqslant j \leqslant n$, $I \models F_j : \mu_j$,

(iii) $I \models F_0 : \mu_0 \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ iff $I \models F_0 : \mu_0$ or $I \not\models (F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$,

(iv) $I \models (\exists x)(F : \mu)$ iff $I \models (F(x/t) : \mu)$ for some ground term t, where $F(x/t)$ denotes the replacement of all free occurrences of x in F by t, and

(v) $I \models (\forall x)(F : \mu)$ iff $I \models (F(x/t) : \mu)$ for all ground terms t.

Note that $I \models F : \mu$ defines a satisfaction relation, that is, a probabilistic interpretation I either satisfies $F : \mu$ or not; I does not try to calculate the probability range for F. As usual, we use the notation $\models$ also to denote logical consequence. We say program P logically entails formula F, denoted $P \models F$ iff whenever I is a probabilistic interpretation that satisfies each clause in P, then $I \models F$. We now investigate the relationship between fixpoints and probabilistic models of a p-program. Lemma 9 below is necessary to prove Theorem 5.

LEMMA 9. *Suppose h is a fully defined formula function. Then for all $F \in bf(B_L)$, $h(F) = [\alpha, \beta]$ is the smallest interval that contains $\{I(F) \mid I \in \mathscr{I}(h)\}$ which is the set of probabilistic truth values of F assigned by the family of probabilistic interpretations associated with h.*

Proof. (i) *Claim.* For all $F \in bf(B_L)$, $h(F) = [\alpha, \beta]$ contains $\{I(F) \mid I \in \mathscr{I}(h)\}$. Suppose there exists an $I \in \mathscr{I}(h)$ such that $I(F) \notin [\alpha, \beta]$. Then $I(F)$ does not satisfy the inequality $\alpha \leqslant (\sum_{K_j \models F \text{ and } K_j \in 2^{B_L}} k_j) = I(F) \leqslant \beta$ as defined in Definition 15. Therefore, I cannot be in $\mathscr{I}(h)$, which is a contradiction!

(ii) *Claim.* For all $F \in bf(B_L)$, if $[\delta, \gamma]$ contains $\{I(F) \mid I \in \mathscr{I}(h)\}$, then $h(F) = [\alpha, \beta] \subseteq [\delta, \gamma]$. Let F be any ground formula in $bf(B_L)$. Proceed by induction on $rank(F)$.

Base Case: $rank(F) = 1$. Then $F \equiv A$ for some ground atom A. Consider some $\mu \in [\alpha, \beta]$. Recall that $\mathscr{K}\mathscr{I}(h) = \mathscr{K}\mathscr{I}(f)$ which is the solution set for $\mathscr{L}\mathscr{P}(f)$ as defined in Definition 15. Construct a new system Q' from $\mathscr{L}\mathscr{P}(f)$ by only replacing the constraint $\alpha \leqslant (\sum_{K_j \models A \text{ and } K_j \in 2^{B_L}} k_j) \leqslant \beta$ with $\mu \leqslant (\sum_{K_j \models A \text{ and } K_j \in 2^{B_L}} k_j) \leqslant \mu$. Recall from Lemma 6 that if h is fully defined, f is fully defined. Since $\varnothing \subset [\mu, \mu] \subseteq [0, 1]$, then by Theorem 4, Q' has a non-empty solution set. That is, there exists a solution KI for Q'. Since $\alpha \leqslant \mu \leqslant \beta$, $KI \in \mathscr{K}\mathscr{I}(f) = \mathscr{K}\mathscr{I}(h)$. Therefore, if I is the probabilistic interpretation corresponding to KI, $I \in \mathscr{I}(h)$. Thus, $I(F) = I(A) = (\sum_{K_j \models A \text{ and } K_j \in 2^{B_L}} k_j) = \mu \in [\delta, \gamma]$. This completes the proof that $[\alpha, \beta] \subseteq [\delta, \gamma]$ for the base case.

Inductive Case: $rank(F) > 1$. Then, F is either a conjunction or a disjunction.

Case 1. $F \equiv C_1 \wedge C_2$. By the induction hypothesis, $h(C_1) = [\alpha_1, \beta_1]$ and $h(C_2) = [\alpha_2, \beta_2]$ are the smallest intervals that contain $\{I(C_1) \mid I \in \mathscr{I}(h)\}$ and $\{I(C_2) \mid I \in \mathscr{I}(h)\}$, respectively. Thus for all $I \in \mathscr{I}(h)$, I satisfies the constraints of the linear program listed in Theorem 1, where $I(C_1) = k_1 + k_2$, $I(C_2) = k_1 + k_3$ and $I(C_1 \wedge C_2) = k_1$. Then by Definition 8, $h(C_1 \wedge C_2) = [\alpha, \beta]$ implies $\alpha = \mathbf{min}_{\mathbf{Q}} I(C_1 \wedge C_2)$ and $\beta = \mathbf{max}_{\mathbf{Q}} I(C_1 \wedge C_2)$. Now suppose there exists $[\delta, \gamma]$ that contains $\{I(C_1 \wedge C_2) \mid I \in \mathscr{I}(h)\}$ such that $[\delta, \gamma] \subset [\alpha, \beta]$. But according to the proof of Theorem 1, there actually exists $I_1, I_2 \in \mathscr{I}(h)$ such that $I_1(C_1 \wedge C_2) = \alpha$ and $I_2(C_1 \wedge C_2) = \beta$. Therefore, $[\delta, \gamma]$ cannot contain both $I_1(C_1 \wedge C_2)$ and $I_2(C_1 \wedge C_2)$, which is a contradiction! Therefore, $[\alpha, \beta] \subseteq [\delta, \gamma]$.

Case 2. $F \equiv D_1 \vee D_2$. The proof is similar to the one for conjunction in case 1. This completes the induction and the proof of the lemma. ∎

The lemma says that $h(F)$ is the smallest interval that constains the set of all the probabilistic truth values of F assigned by the family of probabilistic interpretations associated with h. It is useful in proving the following theorem which states that all fully defined pre-fixpoints of T_P generate a non-empty family of probabilistic models for P.

THEOREM 5. *Suppose a formula function h is fully defined. Then $\mathscr{I}(h)$ is a family of probabilistic models of P iff $T_P(h) \leqslant h$.*

Proof. (i) *Claim.* If h is fully defined, and $\mathscr{I}(h)$ is a family of probabilistic models, then $T_P(h) \leqslant h$.

Let F be any ground formula in $bf(B_L)$. Proceed by induction on $rank(F)$.

Base Case: $rank(F) = 1$. Then $F \equiv A$ for some ground atom A. Consider the atomic function f associated with h. According to Theorem 3, $\mathscr{K}\mathscr{I}(f) = \mathscr{K}\mathscr{I}(h)$. Therefore, $\mathscr{I}(f)$ is a family of probabilistic models. Recall from Definition 11 that for all $A \in B_L$, $T_P(h)(A) = \bigcap \{\mu \mid A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ is a ground instance of a clause in P and $\forall i, 1 \leqslant i \leqslant n$, $h(F_i) \subseteq \mu_i\}$. Let C be any clause described in the set above. Then, for all $I \in \mathscr{I}(f)$, $I(A) \in \mu$, where μ is the p-annotation in the head of C. Therefore, $I(A) \in T_P(h)(A)$. Thus, $T_P(h)(A)$ contains $\{I(A) \mid I \in \mathscr{I}(f)\}$. But by Lemma 9, $f(A)$ is the smallest interval that contains $\{I(A) \mid I \in \mathscr{I}(f)\}$. Therefore, $h(A) = f(A) \subseteq T_P(h)(A)$.

Inductive Case: $rank(F) > 1$. Then, F is either a conjunction or a disjunction.

Case 1. $F \equiv C_1 \wedge C_2$. By the induction hypothesis, $h(C_1) \subseteq T_P(h)(C_1)$ and $h(C_2) \subseteq T_P(h)(C_2)$. But by Lemma 1, $h(C_1 \wedge C_2) = h(C_1) \otimes h(C_2) \subseteq T_P(h)(C_1) \otimes T_P(h)(C_2) = T_P(h)(C_1 \wedge C_2)$.

Case 2. $F \equiv D_1 \vee D_2$. The proof is similar to the one for conjunctions in case 1. This completes the proof of (i).

(ii) *Claim.* If h is fully defined, and $T_P(h) \leqslant h$, then $\mathscr{I}(h)$ is a family of probabilistic models. For all $I \in \mathscr{I}(h)$, let $A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ be a ground instance of a clause in P, and $I(F_i) \in \mu_i$, for all $1 \leqslant i \leqslant n$. But by Lemma 9, $h(F_i)$ is the smallest interval that contains $\{I(F_i) \mid I \in \mathscr{I}(h)\}$. Therefore, $h(F_i) \subseteq \mu_i$, for all $1 \leqslant i \leqslant n$. Recall that $T_P(h)(A) = \bigcap \{\mu \mid A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ is a ground instance of a clause in P and $\forall i, 1 \leqslant i \leqslant n,\ h(F_i) \subseteq \mu_i\}$. Therefore, $T_P(h)(A) \subseteq \mu$. Since $T_P(h) \leqslant h$, $h(A) \subseteq T_P(h)(A) \subseteq \mu$. However, since $I \in \mathscr{I}(h)$, $\alpha \leqslant (\sum_{K_j \models A \text{ and } K_j \in 2^{B_L}} k_j) = I(A) \leqslant \beta$, where $h(A) = [\alpha, \beta]$. Therefore, $I(A) \in \mu$. Thus, $I \models A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$. Combining the results of (i) and (ii), the theorem is proved. ∎

COROLLARY 1. *If h is fully defined and h is a fixpoint of T_P, then $\mathscr{I}(h)$ is a family of probalistic models of P.*

LEMMA 10. *If formula functions h_1 and h_2 are fully defined, then whenever $h_1 \leqslant h_2$, it is necessary that $\mathscr{K}\mathscr{I}(h_1) \supseteq \mathscr{K}\mathscr{I}(h_2)$.*

Proof. Straightforward from Definition 9. ∎

COROLLARY 2. *Suppose the least fixpoint of T_P is fully defined. Then since T_P is monotonic, it is the case that $\mathscr{I}(lfp(T_P)) = \bigcup \{\mathscr{I}(f) \mid T_P(f) \leqslant f\}$.*

According to the two corollaries above, if the least fixpoint of T_P is fully defined, then it generates a non-empty family of probabilistic models for P. Moreover, this family contains the family associated with each pre-fixpoint of T_P. As T_P is a monotone operator on a complete lattice, it is guaranteed to possess at least one fixpoint (and hence there is at least one h such that $T_P(h) \leqslant h$). However, it is possible that there is no formula function h that satisfies each of the conditions below:

- $T_P(h) \leqslant h$ and
- h is fully defined.

For instance, let P be the p-program containing two clauses:

$$q : [0, 0.2] \leftarrow$$

$$q : [0.4, 0.5] \leftarrow .$$

Here, for all $h \in \mathscr{F}\mathscr{F}$, $T_P(h)(q) = \varnothing$, and hence $h \leqslant T_P(h)$ for all $h \in \mathscr{F}\mathscr{F}$. Thus, there is only one atomic function h such that $T_P(h) \leqslant h$, and this is

the function, denoted by h_0, which assigns $\varnothing$ to q, i.e., $h_0(q) = \varnothing$. Clearly, $T_P(h_0) = h_0$. But h_0 is clearly not fully defined. For such a program P and such an h_0, $\mathcal{I}(h) = \varnothing$. In the following, we characterize p-programs whose least fixpoints are fully defined.

DEFINITION 19. A p-program P is *inconsistent* iff it has no probabilistic model.

We now relate probabilistic inconsistency with the notion of full definedness.

LEMMA 11. *Suppose I is a probabilistic model of the* p-*program P. Then for all $F \in bf(B_L)$, if $T_P \uparrow n(F) \subseteq \mu$, it is necessary that $I \models F : \mu$ for all $n \geqslant 0$.*

Proof. Prove by induction on n. ∎

The following theorem demonstrates that there is a close relationship between consistency of probabilistic logic programs and the full definedness of the least fixpoint of the operator associated with the program.

THEOREM 6. *A* p-*program P is inconsistent iff $lfp(T_P)$ is not fully defined.*

Proof. (i) *Claim.* If $lfp(T_P)$ is fully defined, then P has a probabilistic model.

By Theorem 4, $\mathcal{KI}(lfp(T_P))$ is non-empty. Then by Corollary 1, $\mathcal{I}(lfp(T_P))$ is a non-empty family of probabilistic models of P.

(ii) *Claim.* If $lfp(T_P)$ is not fully defined, then P does not have a probabilistic model.

By Lemma 6, if $lfp(T_P)$ is not fully defined, then the atomic function associated with $lfp(T_P)$ is not fully defined. That is, $\exists A \in B_L$ such that $lfp(T_P)(A) = \varnothing$. By Lemma 4, there exists an integer $n \geqslant 0$ such that $T_P \uparrow (n+1)(A) = \varnothing$. Thus there exist clauses $C_1, ..., C_k$ so that for all $1 \leqslant i \leqslant k$, $C_i \equiv A : \rho_i \leftarrow F_1^i : \mu_1^i \wedge \cdots \wedge F_{m_i}^i : \mu_{m_i}^i$, and $T_P \uparrow n(F_j^i) \subseteq \mu_j^i$ for all $1 \leqslant j \leqslant m_i$. In addition $\bigcap_{i=1}^k \rho_i = \varnothing$. Suppose I is a probabilistic model for P and hence $C_1, ..., C_k$. If $I \models A : \rho_i$ (i.e., $I(A) \in \rho_i$) for all $1 \leqslant i \leqslant k$, then $I(A) \in \bigcap_{i=1}^k \rho_i = \varnothing$, which is impossible! Therefore, $\exists i$ such that $I \not\models A : \rho_i$. Since I is a probabilistic model of P, $I \not\models (F_1^i : \mu_1^i \wedge \cdots \wedge F_{m_i}^i : \mu_{m_i}^i)$. Then, there exists a j, $1 \leqslant j \leqslant m_i$, such that $I \not\models F_j^i : \mu_j^i$. However, by the above lemma, since $T_P \uparrow n(F_j^i) \subseteq \mu_j^i$, $I \models F_j^i : \mu_j^i$, which is a contradiction! Therefore, P cannot have a probabilistic model. Combining (i) and (ii) the theorem is proved. ∎

EXAMPLE 6. The following p-program is inconsistent, as $lfp(T_P)(p) = \varnothing$ and $lfp(T_P)(q) = [0.1, 0.6]$:

$$p : [0.3, 0.5] \leftarrow$$

$$q : [0.1, 0.2] \leftarrow p : [0.2, 0.9]$$

$$q : [0.1, 0.6] \leftarrow p : [0.3, 0.4].$$

Example 6 demonstates another distinction between the paraconsistent programs of Blair and Subrahmanian (1987) and the framework proposed here. Blair and Subrahmanian would consider this program to use intervals as truth values. However, in the programs of Blair and Subrahmanian (1987, 1988), every program was guaranteed to possess a model. Moreover, inconsistent theories did not entail all formulas. However, in the probabilistic framework, programs need not possess models, and inconsistent theories entail all formulas. Thus, the system proposed here is not paraconsistent.

Solvability of sets of linear equations has been widely studied (cf. Karwan *et al.*, 1983; Lassez *et al.*, 1989). In our work, linear programs are only used as a tool enabling the model-theoretic study of probabilistic logic programming. As such, we do not address deeper issues in linear programming in this paper.

5. PROOF PROCEDURE

In this section, we show how we may process queries to p-programs.

DEFINITION 20. θ is a *unifier* of annotated conjunctions $C_1 \equiv (A_1 \wedge \cdots \wedge A_n) : \mu_1$ and $C_2 \equiv (B_1 \wedge \cdots \wedge B_m) : \mu_2$ iff $\{A_i\theta \mid 1 \leqslant i \leqslant n\} = \{B_i\theta \mid 1 \leqslant i \leqslant m\}$. Similarly, θ is a *unifier* of annotated disjunctions $D_1 \equiv (A_1 \vee \cdots \vee A_n) : \mu_1$ and $D_2 \equiv (B_1 \vee \cdots \vee B_m) : \mu_2$ iff $\{A_i\theta \mid 1 \leqslant i \leqslant n\} = \{B_i\theta \mid 1 \leqslant i \leqslant m\}$.

In the following we introduce a notion that is analogous to mgu's in the classical paradigm.

DEFINITION 21. Let $UNI(C_1, C_2)$ be the set of unifiers of C_1 and C_2.

(i) Given $\theta_1, \theta_2 \in UNI(C_1, C_2)$, $\theta_1 \leqslant \theta_2$ iff there exists a substitution γ such that $\theta_1 = \theta_2\gamma$.

(ii) $\theta_1 \equiv \theta_2$ iff $\theta_1 \leqslant \theta_2$ and $\theta_2 \leqslant \theta_1$.

Intuitively $\theta_1 \leqslant \theta_2$ means that θ_2 is more general than θ_1. Note that $\leqslant$ may not be a partial ordering, since $\theta_1 \leqslant \theta_2$ and $\theta_2 \leqslant \theta_1$ does not

necessarily imply that $\theta_1 = \theta_2$. It is, however, easy to see that $\equiv$ is an equivalence relation.

DEFINITION 22. Given $\theta \in UNI(C_1, C_2)$, denote the equivalence class of θ by $[\theta]$; i.e., $[\theta] = \{\gamma \mid \theta \equiv \gamma\}$.

(i) $[\theta_1] \leqslant [\theta_2]$ iff there exists γ such that $[\theta_1] = [\theta_2\gamma]$.

(ii) $[\theta_1] < [\theta_2]$ iff $[\theta_1] \leqslant [\theta_2]$ and $[\theta_1] \neq [\theta_2]$.

DEFINITION 23. θ is a *max-gu* (*maximally general unifier*) of C_1 and C_2 iff

(i) θ is a unifier, i.e., $\theta \in UNI(C_1, C_2)$, and

(ii) there does not exist $\theta_1 \in UNI(C_1, C_2)$ such that $[\theta] < [\theta_1]$.

We note here that max-gu's[3] are exactly like mgu's in ordinary logic programming except that they are not necessarily unique.

EXAMPLE 7. Consider the annotated disjunctions below:

$$(p(X, a) \vee p(Y, b)) : [1, 1]$$
$$(p(Z, Z) \vee p(c, W)) : [1, 1].$$

Then the following two substitutions are both max-gu's of the above two annotated disjunctions:

$$\sigma_1 = \{a/X, a/Z, c/Y, b/W\}$$
$$\sigma_2 = \{c/X, a/W, b/Y, b/Z\}.$$

The following lemma guarantees the existence of a max-gu, not necessarily unique, of two basic formulas, if they are unifiable. The proof depends on a result of Martelli and Montanari (1982) relating the unification problem to the solvability of a set of equations of terms.

LEMMA 12. *If two basic formulas are unifiable, then there exists a max-gu of the basic formulas.*

Proof. Here we only show the conjunction case, as the disjunction case is similar. In particular, we show that given $C_1 \equiv (A_1 \wedge \cdots \wedge A_n) : \mu_1$ and $C_2 \equiv (B_1 \wedge \cdots \wedge B_m) : \mu_2$, if C_1 and C_2 are unifiable, then there exists a max-gu of C_1 and C_2. Informally, the proof proceeds as follows. If θ is a

[3] Like the notion of a complete set of mgu's, there is a corresponding notion for max-gu's, which is unique up to the equivalence defined above.

unifier of C_1 and C_2, then for each A_i, $1 \leqslant i \leqslant n$, there must be a $B_{\alpha(i)}$, $1 \leqslant \alpha(i) \leqslant m$, such that $A_i\theta = B_{\alpha(i)}\theta$. Conversely, for each $1 \leqslant j \leqslant m$, there exists a $\beta(j)$, $1 \leqslant \beta(j) \leqslant n$, such that $B_j\theta = A_{\beta(j)}\theta$. Thus, informally, when θ is applied to C_1 and C_2, there is a suitable "match" between the A_i's and the B_j's. We may observe, informally, that C_1 and C_2 are unifiable iff there is a solvable set of equations of the form

$$\{\Lambda_1 = \Gamma_1, ..., \Lambda_r = \Gamma_r\},$$

where each $\Lambda_i \in \{A_1, ..., A_n\}$ and each $\Gamma_j \in \{B_1, ..., B_m\}$ and $\{\Lambda_i \mid 1 \leqslant i \leqslant r\} = \{A_1, ..., A_n\}$ and $\{\Gamma_j \mid 1 \leqslant j \leqslant m\} = \{B_1, ..., B_m\}$. (Note here that we may have the same Λ_i occurring more than once on the left side of an equation, and likewise for the Γ_j's, but the same equation cannot be repeated.) We now give a formal explication of this strategy.

Formal Proof. An atom A is of the form $p(t_1, ..., t_n)$ for some predicate symbol p and some integer $n \geqslant 0$ such that for all $1 \leqslant i \leqslant n$, t_i is a term defined in the usual way. Call n the arity of A. Then let a_i be the arity of A_i for $1 \leqslant i \leqslant n$ and b_j be the arity of B_j for $1 \leqslant j \leqslant m$. Let A_{ik} be the kth argument of A_i for $1 \leqslant k \leqslant a_i$, $1 \leqslant i \leqslant n$ and B_{jk} be the kth argument of B_j for $1 \leqslant k \leqslant b_j$, $1 \leqslant j \leqslant m$. Set up all possible sets of equations such that each set of equations satisfies the following conditions:

(1) The left side of each equation is some A_{ik}, $1 \leqslant k \leqslant a_i$, $1 \leqslant i \leqslant n$.

(2) The right side of each equation is some B_{jk}, $1 \leqslant k \leqslant b_j$, $1 \leqslant j \leqslant m$.

(3) For every equation $A_{ik} = B_{jl}$ for some i, k, j, l, the following conditions are true:

(3.1) A_i and B_j are atoms having the same predicate symbol,

(3.2) $k = l$, and

(3.3) for all $1 \leqslant h \leqslant a_i = b_j$, there exists the equation $A_{ih} = B_{jh}$.

(In other words, whenever equations are set up between the arguments of two atoms, these atoms must have the same predicate symbol, and every argument of the predicate are equated in order.)

(4) For all $1 \leqslant k \leqslant a_i$, $1 \leqslant i \leqslant n$, there exists an equation with A_{ik} on the left side.

(5) For all $1 \leqslant k \leqslant b_j$, $1 \leqslant j \leqslant m$, there exists an equation with B_{jk} on the right side.

(6) No equation is repeated.

Intuitively, as the A_{ik}'s and the B_{jk}'s are arguments of the atoms of the conjunctions, the equations defined above can be viewed as the constraints which the unification must satisfy.

The proof of the lemma then consists of 4 parts.

(i) *Claim.* There are finitely many sets that satisfy the conditions above.

Since all the equations are of the form $A_{ik} = B_{jl}$ for some i, k, j, l such that $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant m$, $1 \leqslant k \leqslant a_i$, $1 \leqslant l \leqslant b_j$, there are at most $nL * mL = nmL^2$ distinct equations where $L = max(\{a_i \mid 1 \leqslant i \leqslant n\} \cup \{b_j \mid 1 \leqslant j \leqslant m\})$. Thus there are at most 2^{nmL^2} sets that satisfy the conditions above. Now let $S_1, ..., S_u$ be all such sets of equations which have solutions. The following shows that since C_1 and C_2 are unifiable, $u \geqslant 1$.

(ii) *Claim.* Any unifier θ of C_1 and C_2 is a solution to some set of equations defined above.

Consider $\theta = \{v_1/t_1, ..., v_h/t_h\}$, where v_i's are variables and t_i's are terms for some h. Rewrite each element of θ in equation form; that is, either as $v_i = t_i$ or $t_i = v_i$ $(1 \leqslant i \leqslant h)$ according to conditions 1 and 2. Consider those A_{ik}'s $(1 \leqslant k \leqslant a_i,\ 1 \leqslant i \leqslant n)$ and B_{jk}'s $(1 \leqslant k \leqslant b_j,\ 1 \leqslant j \leqslant m)$ which do not appear in an equation. They are all ground terms. Since θ is a unifier, each of these ground terms must be matched against itself during unification. Hence, for every A_{ik} $(1 \leqslant k \leqslant a_i,\ 1 \leqslant i \leqslant n)$ or B_{jk} $(1 \leqslant k \leqslant b_j,\ 1 \leqslant j \leqslant m)$ which does not appear in an equation, add the equation $A_{ik} = A_{ik}$ or $B_{jk} = B_{jk}$. Thus the set of equations constructed in this way satisfies conditions 4 and 5. In addition, since θ is a unifier such that $\{A_i\theta \mid 1 \leqslant i \leqslant n\} = \{B_j\theta \mid 1 \leqslant j \leqslant m\}$, the equations constructed satisfy condition 3. Hence this set of equations is the same as S_i for some $1 \leqslant i \leqslant u$. And obviously θ is a solution of this set of equations.

(iii) *Claim.* For all $1 \leqslant i \leqslant u$, a solution to S_i is a unifier.

Given a solution, for every equation where either the left side or the right side is a variable, i.e., $v = t$ or $t = v$, include v/t in θ. Now for any $A_i\theta$ $(1 \leqslant i \leqslant n)$, according to conditions 3 and 4, there exists some $1 \leqslant j \leqslant m$ such that $A_i\theta = B_j\theta$, as θ represents a solution to the equations. Hence $\{A_i\theta \mid 1 \leqslant i \leqslant n\} \subseteq \{B_j\theta \mid 1 \leqslant j \leqslant m\}$. Similarly, for every $B_j\theta$ $(1 \leqslant j \leqslant m)$, according to conditions 3 and 5, there exists some $1 \leqslant i \leqslant n$ such that $A_i\theta = B_j\theta$. Hence $\{A_i\theta \mid 1 \leqslant i \leqslant n\} \supseteq \{B_j\theta \mid 1 \leqslant j \leqslant m\}$. Therefore, $\{A_i\theta \mid 1 \leqslant i \leqslant n\} = \{B_j\theta \mid 1 \leqslant j \leqslant m\}$. Thus θ is a unifier.

(iv) *Claim.* There exists a max-gu of C_1 and C_2.

Recall that $S_1, ..., S_u$ are the sets of equations that have solutions. From (ii) and (iii), it is proved that solutions to $S_1, ..., S_u$ correspond to all the unifiers of C_1 and C_2. Now for $1 \leqslant i \leqslant u$, as S_i is solvable, it follows by the result of Martelli and Montanari (1982) that each S_i has an mgu θ_i. Moreover, θ_i is unique in the sense that if σ_i is also an mgu of S_i, then $\theta_i \equiv \sigma_i$; i.e., $[\theta_i] = [\sigma_i]$. As we need to consider only finitely many such θ_i's, it follows that $\{[\theta_1], ..., [\theta_u]\}$ contain a maximal element (again not

necessarily unique) w.r.t. the $\leqslant$ ordering. This maximal element is a max-gu of C_1 and C_2. ▮

The proof of the above lemma yields an algorithm to compute max-gu's (or determine their non-existence). We believe that more efficient algorithms exist, but the study of such algorithms is beyond the scope of this paper. Once again, the main reason that we distinguish between max-gu's and ordinary mgu's is that the former are not always unique. In the remainder of this section, we present a proof procedure for p-programs. As this procedure operates on the compiled version of p-programs, we first formalize the compilation process.

DEFINITION 24. Given a p-program P, define $REDUN(P) = P \cup \{A: [0, 1] \leftarrow \mid A \in B_L\}$.

DEFINITION 25. (1) Given a pair of *distinct* clauses C_1, C_2 of the form $C_1 \equiv A_1 : \mu_1 \leftarrow Body_1$ and $C_2 \equiv A_2 : \mu_2 \leftarrow Body_2$ such that A_1, A_2 are unifiable via max-gu θ, let the clause R_{C_1, C_2} be the clause $A_1\theta : (\mu_1 \cap \mu_2) \leftarrow (Body_1 \wedge Body_2)\,\theta$.

(2) The *closure* of a p-program P, denoted by $CL(P)$, is the p-program constructed by repeatedly adding to P all the clauses R_{C_1, C_2} obtained from distinct p-clauses C_1 and C_2 in P whose heads are unifiable.

Note that in generating the closure of a p-program, unlike the treatment in Blair and Subrahmanian (1987, 1988), it is sufficient only to consider pairs of distinct clauses in P, instead of triplets, quadruplets, and so on. Suppose there is a clause $R_{C_1, \ldots, C_n}$ that is generated from clauses $C_1, \ldots, C_n$ in the way defined above, where $n \geqslant 2$. But observe that for $n \geqslant 2$ and $\mu_i = [\alpha_i, \beta_i]$ for all $1 \leqslant i \leqslant n$, $\mu_1 \cap \cdots \cap \mu_n = [\alpha_1, \beta_1] \cap \cdots \cap [\alpha_n, \beta_n] = [max_{i=1}^{n}\, \alpha_i, min_{j=1}^{n}\, \beta_j] = \mu_{i_0} \cap \mu_{j_0}$, for some i_0, j_0 where $1 \leqslant i_0, j_0 \leqslant n$. Thus, the clause $R_{C_{i_0}, C_{j_0}}$ can replace $R_{C_1, \ldots, C_n}$.

DEFINITION 26. Given a p-program P whose clauses are all standardized apart (cf. Lloyd, 1987), let m be the number of clauses in P. Then the *normal form* of P, denoted by $NF(P)$, is defined as follows:

(i) $CF_1(P) = DF_1(P) = CL(REDUN(P))$

(ii) for all $2 \leqslant i \leqslant m$,

$$CF_i(P) = \{(A_1 \wedge \cdots \wedge A_i) : \mu \leftarrow Body_1 \wedge \cdots \wedge Body_i \mid \forall 1 \leqslant j \leqslant i,$$
$$A_j : \mu_j \leftarrow Body_j \in CL(REDUN(P)), \text{ and } \mu = \mu_1 \otimes \cdots \otimes \mu_i,$$
$$\text{and } \forall 1 \leqslant k, l \leqslant i, k \neq l \Rightarrow A_k \neq A_l\},$$

$$DF_i(P) = \{(A_1 \vee \cdots \vee A_i) : \mu \leftarrow Body_1 \wedge \cdots \wedge Body_i \mid \forall 1 \leqslant j \leqslant i,$$
$$A_j : \mu_j \leftarrow Body_j \in CL(REDUN(P)), \text{ and } \mu = \mu_1 \oplus \cdots \oplus \mu_i,$$
$$\text{and } \forall 1 \leqslant k, l \leqslant i, k \neq l \Rightarrow A_k \neq A_l\}$$

(iii) $NF(P) = \bigcup_{i=1}^{m} (CF_i(P) \cup DF_i(P))$

EXAMPLE 8. Let $P = \{B_1 : \mu_1 \leftarrow Body_1,\ B_1 : \mu_2 \leftarrow Body_2,\ B_2 : \mu_3 \leftarrow Body_3\}$. Then:

(1) $REDUN(P) = P \cup \{B_1 : [0, 1] \leftarrow,\ B_2 : [0, 1] \leftarrow\}$.

(2) $CL(REDUN(P)) = REDUN(P) \cup \{$
$B_1 : (\mu_1 \cap \mu_2) \leftarrow Body_1 \wedge Body_2,$
$B_1 : (\mu_1 \cap [0, 1]) \leftarrow Body_1,$
$B_1 : (\mu_2 \cap [0, 1]) \leftarrow Body_2,$
$B_2 : (\mu_3 \cap [0, 1]) \leftarrow Body_3\}$
$= REDUN(P) \cup \{B_1 : (\mu_1 \cap \mu_2) \leftarrow Body_1 \wedge Body_2\}$.

(3) $CF_2(P) = \{(B_1 \wedge B_2) : [0, 1] \leftarrow,$
$(B_1 \wedge B_2) : (\mu_1 \otimes [0, 1]) \leftarrow Body_1,$
$(B_1 \wedge B_2) : (\mu_2 \otimes [0, 1]) \leftarrow Body_2,$
$(B_1 \wedge B_2) : ((\mu_1 \cap \mu_2) \otimes [0, 1]) \leftarrow Body_1 \wedge Body_2,$
$(B_1 \wedge B_2) : ([0, 1] \otimes \mu_3) \leftarrow Body_3,$
$(B_1 \wedge B_2) : (\mu_1 \otimes \mu_3) \leftarrow Body_1 \wedge Body_3,$
$(B_1 \wedge B_2) : (\mu_2 \otimes \mu_3) \leftarrow Body_2 \wedge Body_3,$
$(B_1 \wedge B_2) : ((\mu_1 \cap \mu_2) \otimes \mu_3) \leftarrow Body_1 \wedge Body_2 \wedge Body_3\}$.

(4) $DF_2(P)$ can be obtained in a similar way.

(5) Finally, $DF_3(P) = CF_3(P) = \varnothing$.

We now show that the clauses added to P to construct $NF(P)$ are logical consequences of P, and hence, they do not change the meaning of P. The addition of such clauses is to ensure that the truth value assigned to any basic formula depends on a single clause, rather than a group of such clauses.

LEMMA 13. *For every clause* $C \in NF(P)$, $P \models C$.

Proof. Case 1. $C \in P$. Then if I is a probabilistic model of P, I must be a probabilistic model of C. Therefore, $P \models C$.

Case 2. $C \in REDUN(P) - P$. Then $C \equiv A : [0, 1] \leftarrow$ for some $A \in B_L$. But for any probabilistic model I of P, $I(A) \in [0, 1]$. Thus $I \models A : [0, 1]$. Therefore, $P \models C$.

Case 3. $C \in CL(REDUN(P)) - REDUN(P)$. Then $C \equiv A\theta : (\mu_1 \cap \mu_2) \leftarrow (Body_1 \wedge Body_2)\theta$. Recall from Definition 25 that the clause C is

derived from two clauses in $REDUN(P)$, namely $A_1 : \mu_1 \leftarrow Body_1$ and $A_2 : \mu_2 \leftarrow Body_2$, such that A_1, A_2 are unifiable via max-gu θ. (We assume that all clauses in $CL(REDUN(P))$ are standardized apart.) Let $C\theta' \equiv A\theta\theta' : (\mu_1 \cap \mu_2) \leftarrow (Body_1 \wedge Body_2)\,\theta\theta'$ be a ground instance of C. Suppose I is a probabilistic model of P and therefore $REDUN(P)$, as shown in case 2 above, and $I \models (Body_1 \wedge Body_2)\,\theta\theta'$. Then $I \models Body_1\,\theta\theta'$ and $I \models Body_2\,\theta\theta'$. Since I is a probabilistic model of $REDUN(P)$, $I \models A\theta\theta' : \mu_1$ and $I \models A\theta\theta' : \mu_2$. Hence it follows that $I \models A\theta\theta' : (\mu_1 \cap \mu_2)$. Thus, $I \models C\theta'$, for any substitution θ'. Thus $I \models C$.

Case 4. $C \in \bigcup_{i=2}^{l} CF_i(P)$, where l is the cardinality of P. Then for some $2 \leqslant k \leqslant l$, $C \equiv (A_1 \wedge \cdots \wedge A_k) : \mu \leftarrow Body_1 \wedge \cdots \wedge Body_k$ such that $\forall 1 \leqslant j \leqslant k$, $A_j : \mu_j \leftarrow Body_j \in CL(REDUN(P))$, $\mu = \mu_1 \otimes \cdots \otimes \mu_k$, $\forall 1 \leqslant m, n \leqslant k$, $m \neq n \Rightarrow A_m \neq A_n$. Let $C\theta' \equiv (A_1 \wedge \cdots \wedge A_k)\,\theta' : \mu \leftarrow (Body_1 \wedge \cdots \wedge Body_k)\,\theta'$ be a ground instance of C. Let I be a probabilistic model of P and therefore $CL(REDUN(P))$, as shown in cases 1, 2 and 3 above. Further suppose that $I \models (Body_1 \wedge \cdots \wedge Body_k)\,\theta'$. Then for all $1 \leqslant j \leqslant k$, $I \models Body_j\,\theta'$. But since I is a probabilistic model of P and $CL(REDUN(P))$, then for all $1 \leqslant j \leqslant k$, $I \models A_j\theta' : \mu_j$; i.e., $I(A_j\theta') \in \mu_j$. But according to Theorem 1, $I(A_1\theta' \wedge A_2\theta') \in \mu_1 \otimes \mu_2$. By applying the same theorem repeatedly, $I((A_1 \wedge \cdots \wedge A_k)\,\theta') \in \mu_1 \otimes \cdots \otimes \mu_k = \mu$. Therefore, $I \models (A_1 \wedge \cdots \wedge A_k)\,\theta' : \mu$. That is, $I \models C\theta'$, for any substitution θ'. Thus, $I \models C$.

Case 5. $C \in \bigcup_{i=2}^{l} DF_i(P)$, where l is the cardinality of P. The proof is similar to the one for conjunctions in Case 4. This completes the proof for the lemma. ∎

We now present a refutation procedure for query processing.

DEFINITION 27. A *query* is a formula of the form $\exists(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ where for all $1 \leqslant i \leqslant n$, F_i is a basic formula, not necessarily ground.

DEFINITION 28. Suppose $C \equiv G_0 : \lambda_0 \leftarrow G_1 : \lambda_1 \wedge \cdots \wedge G_m : \lambda_m$ is a clause in $NF(P)$ and $Q \equiv \exists(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ is a query, and Q, C are standardized apart. Then

$$\exists((F_1 : \mu_1 \wedge \cdots \wedge F_{i-1} : \mu_{i-1} \wedge G_1 : \lambda_1 \wedge \cdots \wedge G_m : \lambda_m \wedge F_{i+1} : \mu_{i+1} \wedge \cdots \wedge F_n : \mu_n)\,\theta)$$

is an *SLDp-resolvent* of C and Q on $F_i : \mu_i$ iff

(i) θ is a max-gu of G_0 and F_i, and

(ii) $\lambda_0 \subseteq \mu_i$.

If θ is a unifier, but not necessarily a max-gu, then the resolvent is called an *unrestricted SLDp-resolvent.*

DEFINITION 29. An *SLDp-deduction* of the initial query $Q_1 \equiv \exists(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ from a p-program P is a sequence $\langle Q_1, C_1, \theta_1 \rangle, ..., \langle Q_r, C_r, \theta_r \rangle, ...$, where for all $i \geq 1$, C_i is a renamed version of a clause in $NF(P)$ and Q_{i+1} is an SLDp-resolvent of Q_i and C_i via a max-gu θ_i.

If the θ_i's are not restricted to be max-gu's, then the sequence is called an *unrestricted SLDp-deduction.*

DEFINITION 30. An *SLDp-refutation* of the initial query $Q_1 \equiv \exists(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ from a p-program P is a finite SLDp-deduction $\langle Q_1, C_1, \theta_1 \rangle, ..., \langle Q_n, C_n, \theta_n \rangle$, where the SLDp-resolvent of Q_n and C_n via max-gu θ_n is the empty query. $\theta_1 ... \theta_n$ is called the *computed answer substitution.* If the θ_i's are not restricted to be max-gu's, then the deduction is called an *unrestricted SLDp-refutation.*

We now demonstrate that SLDp-refutation is always sound. Hereafter, given a query $Q \equiv \exists(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ and a substitution θ, we abuse the notation $\forall(Q\theta)$ to denote $\forall(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)\,\theta$.

THEOREM 7 (Soundness of SLDp-refutation). *If there exists an SLDp-refutation of the initial query* $Q_1 \equiv \exists(F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$ *from a* p*-program* P, *then* $P \models \forall(Q_1\theta)$, *where* θ *is the computed answer substitution.*

Proof. Let the given SLDp-refutation be $\langle Q_1, C_1, \theta_1 \rangle, ..., \langle Q_n, C_n, \theta_n \rangle$. Proceed by induction on n, the length of the refutation.

Base Case: $n = 1$. Then $Q_1 \equiv F_1 : \mu_1$ and $C_1 \equiv G_0 : \lambda_0 \leftarrow$ is a renamed version of a clause in $NF(P)$ such that $F_1\theta_1 = G_0\theta_1$ and $\lambda_0 \subseteq \mu_1$. Suppose I is a probabilistic model of P. Then by Lemma 13, I is a probabilistic model of all clauses in $NF(P)$. Therefore, $I \models \forall(G_0 : \lambda_0)$. In particular, $I \models \forall((G_0 : \lambda_0)\,\theta_1)$. That is, $I \models \forall((G_0\theta_1) : \lambda_0)$. Thus, $I \models \forall((G_0\theta_1) : \mu_1)$ as $\lambda_0 \subseteq \mu_1$. That is, $I \models \forall((F_1\theta_1) : \mu_1)$. Thus, $I \models \forall((F_1 : \mu_1)\,\theta_1)$; i.e., $I \models \forall(Q_1\theta_1)$.

Inductive Case: $n > 1$. Let $Q_1 \equiv (F_1 : \mu_1 \wedge \cdots \wedge F_m : \mu_m)$ and $C_1 \equiv G_0 : \lambda_0 \leftarrow Body_1$ be a renamed version of a clause in $NF(P)$ such that $F_i\theta_1 = G_0\theta_1$ and $\lambda_0 \subseteq \mu_i$. Now $Q_2 \equiv (F_1 : \mu_1 \wedge \cdots \wedge F_{i-1} : \mu_{i-1} \wedge Body_1 \wedge F_{i+1} : \mu_{i+1} \wedge \cdots \wedge F_m : \mu_m)\,\theta_1$. But by the induction hypothesis, $P \models \forall(Q_2\theta_2 \cdots \theta_n)$. In other words, $P \models \forall((F_1 : \mu_1 \wedge \cdots \wedge F_{i-1} : \mu_{i-1} \wedge Body_1 \wedge F_{i+1} : \mu_{i+1} \wedge \cdots \wedge F_m : \mu_m)\,\theta_1\theta_2 \cdots \theta_n)$. Therefore, $P \models \forall((F_1 : \mu_1 \wedge \cdots \wedge F_{i-1} : \mu_{i-1} \wedge F_{i+1} : \mu_{i+1} \wedge \cdots \wedge F_m : \mu_m)\,\theta_1 \cdots \theta_n)$, and $P \models \forall((Body_1)\,\theta_1 \cdots \theta_n)$. From the latter, it follows that $P \models \forall((G_0\theta_1 \cdots \theta_n) : \lambda_0)$.

Therefore, as $F_i\theta_1 = G_0\theta_1$, it follows that $P \models \forall((F_i\theta_1 \cdots \theta_n) : \lambda_0)$. Since $\lambda_0 \subseteq \mu_i$, $P \models \forall((F_i\theta_1 \cdots \theta_n) : \mu_i) = \forall((F_i : \mu_i)\,\theta_1 \cdots \theta_n)$. Thus, $P \models \forall((F_1 : \mu_1 \wedge \cdots \wedge F_{i-1} : \mu_{i-1} \wedge F_i : \mu_i \wedge F_{i+1} : \mu_{i+1} \wedge \cdots \wedge F_m : \mu_m)\,\theta_1 \cdots \theta_n)$. Therefore, $P \models \forall(Q_1\theta_1 \cdots \theta_n)$, i.e. $P \models \forall(Q_1\theta)$. ∎

Before we proceed to prove the completeness of SLDp-refutation, we prove several lemmas below, all of which are used in proving Theorem 8 (the Completeness Theorem).

LEMMA 14 (Max-gu Lemma). *Suppose query Q_1 has an unrestricted SLDp-refutation from P. Then Q_1 has an SLDp-refutation from P of the same length, such that if $\theta_1, ..., \theta_n$ are the unifiers in the unrestricted refutation, and $\theta'_1, ..., \theta'_n$ are the max-gu's used in the SLDp-refutation, then $\theta_1 \cdots \theta_n = \theta'_1 \cdots \theta'_n\gamma$ for some γ.*

Proof. Similar to the proof for the classical case (Lloyd, 1987). ∎

LEMMA 15 (Lifting Lemma). *Let Q_1 be a query and θ be a substitution. Suppose $Q_1\theta$ has an SLDp-refutation from P. Then Q_1 has an SLDp-refutation from P of the same length. Moreover, if $\theta_1, ..., \theta_n$ are the max-gu's used in the SLDp-refutation from $Q_1\theta$, and $\theta'_1, ..., \theta'_n$ are the max-gu's used in the SLDp-refutation from Q_1, then $\theta\theta_1 \cdots \theta_n = \theta'_1 \cdots \theta'_n\gamma$ for some γ.*

Proof. Similar to the proof for the classical case (Lloyd, 1987). ∎

Lemma 16 below requires that P be a consistent p-program.

LEMMA 16. (1) *Let P be a consistent* p*-program, and $C_1, C_2 \in conj(B_L)$. Suppose $P \models (C_1 \wedge C_2) : \mu$. Then $P \models C_1 : \mu_1$ and $P \models C_2 : \mu_2$ for some μ_1, μ_2 such that $\mu \supseteq \mu_1 \otimes \mu_2$.*

(2) *Similarly, given $D_1, D_2 \in disj(B_L)$, if $P \models (D_1 \vee D_2) : \mu$, then $P \models D_1 : \mu_1$ and $P \models D_2 : \mu_2$ for some μ_1, μ_2 such that $\mu \supseteq \mu_1 \oplus \mu_2$.*

Proof. Since P is consistent, then for every probabilistic model I of P, $I(C_1 \wedge C_2) \in \mu$. Recall from Corollary 2 that the family of probabilistic models corresponding to the least fixpoint contains all probabilistic models for P. Hence, it follows from Lemma 9 that $lfp(T_P)(C_1 \wedge C_2) \subseteq \mu$. But $lfp(T_P)(C_1 \wedge C_2) = lfp(T_P)(C_1) \otimes lfp(T_P)(C_2)$. Let μ_1 and μ_2 be $lfp(T_P)(C_1)$ and $lfp(T_P)(C_2)$, respectively. In other words, $\mu_1 \otimes \mu_2 \subseteq \mu$. Again, by Lemma 9, for every probabilistic model I of P, $I(C_1) \in \mu_1$ and $I(C_2) \in \mu_2$. Therefore, $P \models C_1 : \mu_1$ and $P \models C_2 : \mu_2$. This completes the proof of (1). The proof of (2) is similar. ∎

LEMMA 17. *Let P be a p-program, F be a basic formula (not necessarily ground), and $\mu \subseteq [0, 1]$. Then $P \models \exists(F : \mu) \Rightarrow P \models F\theta : \mu$ for some ground instance $F\theta$ of F.*

Proof. Proceed by induction on $rank(F)$.

Base Case: $rank(F) = 1$. Then $F \equiv A$ for some atom A. Let $\{A_1, ..., A_m\}$ be all the ground instances of A, and $lfp(T_P)$ be denoted by h. Further suppose that for all $1 \leqslant i \leqslant m$, $h(A_i) = [\alpha_i, \beta_i]$. Suppose there exists no i such that $1 \leqslant i \leqslant m$ and $[\alpha_i, \beta_i] \subset \mu$. That is, for all $1 \leqslant i \leqslant m$, $\mu \not\supseteq [\alpha_i, \beta_i]$. In other words, for all $1 \leqslant i \leqslant m$, there exists $\gamma_i \in [\alpha_i, \beta_i]$ such that $\gamma_i \notin \mu$. Now construct a linear program Q from $\mathscr{LP}(h)$ by replacing the m linear inequalities of the form $\alpha_i \leqslant (\sum_{K_j \models A_i \text{ and } K_j \in 2^{B_L}} k_j) \leqslant \beta_i$ with $\gamma_i \leqslant (\sum_{K_j \models A_i \text{ and } K_j \in 2^{B_L}} k_j) \leqslant \gamma_i$, for $1 \leqslant i \leqslant m$. Then by Theorem 4, the solution set of Q is non-empty. That is, there exists a solution KI for Q. Since $\alpha_i \leqslant \gamma_i \leqslant \beta_i$, for all $1 \leqslant i \leqslant m$, $KI \in \mathscr{KI}(h)$. (Recall from Definition 15 that $\mathscr{KI}(h)$ is the family of kernel interpretations corresponding to h.) Therefore, if J is the probabilistic interpretation constructed from $\mathscr{KI}$, $J \in \mathscr{I}(h)$. Thus, J is a probabilistic model of P. But for all $1 \leqslant i \leqslant m$, $J(A_i) = \gamma_i \notin \mu$. Therefore, $J \not\models \exists(A : \mu)$, which is a contradiction! Therefore, there exists an i such that $1 \leqslant i \leqslant m$ and $[\alpha_i, \beta_i] \subseteq \mu$. Now for any probabilistic model I of P, $I \models A_i : [\alpha_i, \beta_i]$. Therefore, $I \models A_i : \mu$. Thus, it follows that $P \models A_i : \mu$.

Inductive Case: $rank(F) > 1$. Then F is either a conjunction or a disjunction.

Case 1. $F \equiv C \wedge D$. Let $\{F_1, ..., F_k\}$ be all the ground instances of F. For each $1 \leqslant i \leqslant k$, there exists a substitution θ_i such that $F_i \equiv F\theta_i \equiv C\theta_i \wedge D\theta_i$. Let h be $lfp(T_P)$. Then suppose it is not true that there exists i such that $1 \leqslant i \leqslant k$ and $\rho_i \otimes \lambda_i \subseteq \mu$, where for all $1 \leqslant i \leqslant k$, $h(C\theta_i) = \rho_i$ and $h(D\theta_i) = \lambda_i$. In other words, for all $1 \leqslant i \leqslant k$, $\rho_i \otimes \lambda_i \not\subseteq \mu$. Let $\rho_i \otimes \lambda_i = [\delta_i^l, \delta_i^u]$ and $(\rho_i \otimes \lambda_i) \cap \mu = [\gamma_i^l, \gamma_i^u]$, for all $1 \leqslant i \leqslant k$. Then, by our assumption, $[\gamma_i^l, \gamma_i^u]$ cannot contain both δ_i^l and δ_i^u at the same time, for all $1 \leqslant i \leqslant k$. Now consider the following set of linear inequalities Q on any probabilistic model I:

$$\begin{array}{llll} \text{for all} & 1 \leqslant i \leqslant k, & \rho_i^l \leqslant I(C\theta_i) \leqslant \rho_i^u, & \text{where} \quad \rho_i = [\rho_i^l, \rho_i^u] \\ \text{for all} & 1 \leqslant i \leqslant k, & \lambda_i^l \leqslant I(D\theta_i) \leqslant \lambda_i^u, & \text{where} \quad \lambda_i = [\lambda_i^l, \lambda_i^u] \\ \text{for all} & 1 \leqslant i \leqslant k, & \delta_i^l \leqslant I(C\theta_i \wedge D\theta_i) \leqslant \delta_i^u. & \end{array}$$

Consider the new set of linear inequalities Q' constructed from Q (that is, replace the ranges $\rho_i \otimes \lambda_i$ by $(\rho_i \otimes \lambda_i) \cap \mu$, for all $1 \leqslant i \leqslant k$):

$$\begin{array}{lll} \text{for all} & 1 \leqslant i \leqslant k, & \rho_i^l \leqslant I(C\theta_i) \leqslant \rho_i^u \\ \text{for all} & 1 \leqslant i \leqslant k, & \lambda_i^l \leqslant I(D\theta_i) \leqslant \lambda_i^u \\ \text{for all} & 1 \leqslant i \leqslant k, & \gamma_i^l \leqslant I(C\theta_i \wedge D\theta_i) \leqslant \gamma_i^u. \end{array}$$

But according to Lemma 9, all ranges computed by h are the tightest possible ranges. This is because, as shown in the proof of Lemma 9, probabilistic interpretations can always be constructed to take on the upper or lower bounds of ranges computed by h for all basic formulas. Then from the induction hypothesis, there exist probabilistic models that satisfy the following inequalities:

$$\text{for all} \quad 1 \leqslant i \leqslant k, \qquad \rho_i^l \leqslant I(C\theta_i) \leqslant \rho_i^u$$

$$\text{for all} \quad 1 \leqslant i \leqslant k, \qquad \lambda_i^l \leqslant I(D\theta_i) \leqslant \lambda_i^u.$$

In particular, there are models that take on the upper and lower bounds of the inequalities. Thus according to Theorem 1, these models satisfy system Q above. However, since for all $1 \leqslant i \leqslant k$, $\rho_i \otimes \lambda_i \not\subseteq \mu$, either δ_i^l or δ_i^u does not lie in $[\gamma_i^l, \gamma_i^u]$. Therefore, there exists a probabilistic model I such that it satisfies system Q above, but for all $1 \leqslant i \leqslant k$, $I(C\theta_i \wedge D\theta_i)$ does not lie in μ. Thus, $I \not\models \exists(C \wedge D) : \mu$, which is a contradiction! Therefore, there exists an i such that $1 \leqslant i \leqslant k$ and $\rho_i \otimes \lambda_i \subseteq \mu$. Now for any probabilistic model I of P, $I \models (C\theta_i \wedge D\theta_i) : (\rho_i \otimes \lambda_i)$. Therefore, $I \models (C\theta_i \wedge D\theta_i) : \mu$. Thus, it follows that $P \models (C \wedge D)\theta_i : \mu$.

Case 2. $F \equiv C \vee D$. The proof is similar to the one for conjunctions in case 1. This completes the induction and the proof of the lemma. ∎

LEMMA 18. *Let P be a probabilistic program and $Q \equiv (F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n)$. If $P \models \exists Q$, then $P \models Q\theta$ for some ground instance $Q\theta$ of Q.*

Proof. Let $\{Q_1, ..., Q_m\}$ be all the ground instances of Q. (Recall that our language is function free, and thus there are only finitely many such ground instances.) For each $1 \leqslant j \leqslant m$, there exists a substitution θ_j such that $Q_j \equiv Q\theta_j \equiv F_1\theta_j : \mu_1 \wedge \cdots \wedge F_n\theta_j : \mu_n$. Abbreviate $lfp(T_P)$ by h. Let for all $1 \leqslant j \leqslant m$, $1 \leqslant i \leqslant n$, $h(F_i\theta_j) = [\rho_{ij}^l, \rho_{ij}^u]$. Assume that there does not exist a θ_k such that $1 \leqslant k \leqslant m$ and for all $1 \leqslant i \leqslant n$, $[\rho_{ik}^l, \rho_{ik}^u] \subseteq \mu_i$. In other words, for all $1 \leqslant j \leqslant m$, there exists $1 \leqslant i \leqslant n$, such that $[\rho_{ij}^l, \rho_{ij}^u] \not\subseteq \mu_i$. Let for all $1 \leqslant j \leqslant m$, $1 \leqslant i \leqslant n$, $[\rho_{ij}^l, \rho_{ij}^u] \cap \mu_i = [\gamma_{ij}^l, \gamma_{ij}^u]$. Then by our assumption, for all $1 \leqslant j \leqslant m$, there exists $1 \leqslant i \leqslant n$, such that $[\gamma_{ij}^l, \gamma_{ij}^u]$ cannot contain both ρ_{ij}^l and ρ_{ij}^u. Recall that for any probabilistic model I of P, I satisfies the following set Q of linear inequalities:

$$\text{for all } 1 \leqslant i \leqslant n, \qquad \rho_{i1}^l \leqslant I(F_i\theta_1) \leqslant \rho_{i1}^u$$

$$\cdots$$

$$\text{for all } 1 \leqslant i \leqslant n, \qquad \rho_{im}^l \leqslant I(F_i\theta_m) \leqslant \rho_{im}^u.$$

Now consider the new set of linear inequalities Q' constructed from Q (that is, replace the ranges $[\rho_{ij}^l, \rho_{ij}^u]$ with $[\rho_{ij}^l, \rho_{ij}^u] \cap \mu_i = [\gamma_{ij}^l, \gamma_{ij}^u]$ for all $1 \leqslant j \leqslant m$, $1 \leqslant i \leqslant n$):

$$\text{for all } 1 \leqslant i \leqslant n, \qquad \gamma_{i1}^l \leqslant I(F_i\theta_1) \leqslant \gamma_{i1}^u$$

$$\cdots$$

$$\text{for all } 1 \leqslant i \leqslant n, \qquad \gamma_{im}^l \leqslant I(F_i\theta_m) \leqslant \gamma_{im}^u.$$

But according to Lemma 9, all ranges computed by h are the tightest possible ranges. This is because, as shown in the proof of Lemma 9, probabilistic models can always be constructed to take on the upper or lower bounds of ranges computed by h for all basic formulas. In particular, there exist models I of P such that $I(F_i\theta_j)$ takes on the values either ρ_{ij}^l or ρ_{ij}^u, for all $1 \leqslant j \leqslant m$ and some $1 \leqslant i \leqslant n$ dependent on j. Therefore, there exists a model I of P such that $I(F_i\theta_j) \notin \mu_i$ for all $1 \leqslant j \leqslant m$ and some $1 \leqslant i \leqslant n$ dependent on j. Thus, for all $1 \leqslant j \leqslant m$, $I \not\models F_1\theta_j : \mu_1 \wedge \cdots \wedge F_n\theta_j : \mu_n$, which contradicts the assumption that $P \models \exists Q$! Therefore, there exists a θ_k such that $1 \leqslant k \leqslant m$ and for all $1 \leqslant i \leqslant n$, $[\rho_{ij}^l, \rho_{ij}^u] \subseteq \mu_i$. Now recall that for any probabilistic model I of P, $I \models F_i\theta_k : [\rho_{ik}^l, \rho_{ik}^u]$, for all $1 \leqslant i \leqslant n$. Thus, $I \models F_i\theta_k : \mu_i$, for all $1 \leqslant i \leqslant n$. Therefore, it follows that $I \models Q\theta_k$ where I is any probabilistic model of P. Thus, $P \models Q\theta_k$. ∎

Lemmas 17 and 18 may seem false when one consider $F_i \in disj(B_L)$. Consider the classical logic sentence $p(a) \vee p(b)$. This sentence may be represented as the single clause p-program P:

$$(p(a) \vee p(b)) : [1, 1] \leftarrow .$$

Note that P does *not* probabilistically entail the query $(\exists x)(p(x) : [1, 1])$. To see this, let KI be the probabilistic kernel interpretation that assigns $\frac{1}{3}$ to each of the worlds $K_1 = \{p(a), p(b)\}$, $K_2 = \{p(a)\}$, $K_3 = \{p(b)\}$ and 0 to $K_4 = \varnothing$. Let I be the probabilistic interpretation associated with KI. Then I is a probabilistic model of P, but I assigns $\frac{2}{3}$ to each of $p(a), p(b)$. Thus, $I \not\models (\exists x)(p(x) : [1, 1]))$. However, I assigns 1 to $(\exists x)\, p(x)$. This is probabilistically correct because, in general, the probabilistic statement $\mathbf{P}((\exists x)\, q(x)) = v$ is *not* equivalent to the (metalinguistic) statement $(\exists x)(\mathbf{P}(q(x)) = v)$.

LEMMA 19. *Let P be a consistent* p*-program, F be a basic formula, and θ be a substitution. For $\alpha > 1$, if $T_P \uparrow \alpha(F\theta) \subseteq \mu$, then there exists a clause C in $NF(P)$ having an instance of the form $G : \mu' \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ such that $\mu' \subseteq \mu$ and for all $1 \leqslant i \leqslant n$, $T_P \uparrow (\alpha - 1)(F_i) \subseteq \mu_i$ and G subsumes $F\theta$.*

Proof. Proceed by induction on $rank(F)$.

Base Case: $rank(F) = 1$. Then $F \equiv A$ for some atom A. Recall from Definition 11 that $T_P \uparrow \alpha(A) = \bigcap S$, where $S = \{\mu \mid A : \mu \leftarrow F_1 : \mu_1 \wedge \cdots \wedge F_n : \mu_n$ is a ground instance of a clause in P and $\forall i, 1 \leqslant i \leqslant n$, $T_P \uparrow (\alpha - 1)(F_i) \subseteq \mu_i\}$. If $S = \varnothing$, then $T_P \uparrow \alpha(A) = \bot = [0, 1]$. But recall that the clause $A : [0, 1] \leftarrow$ is in $REDUN(P)$ and is therefore in $NF(P)$. So this clause satisfies the requirement of the lemma. If there is only one element μ in S, then the clause whose head annotation is μ is in P and therefore in $NF(P)$. The lemma is therefore proved for this case. If however there is more than one μ in S, then there exists a pair $\mu_i, \mu_j \in S$ such that $\mu_i \cap \mu_j = \bigcap S$. But then by Definition 25, there exists a clause R_{C_i, C_2} in $CL(REDUN(P))$ and therefore in $NF(P)$ such that it is the closure of the two clauses C_i, C_j whose head annotations are μ_i, μ_j, respectively, and whose head's atomic parts have A as a common instance. And this clause satisfies the requirement of the lemma.

Inductive Case: $rank(F) > 1$. Then F is either a conjunction or a disjunction.

Case 1. $F \equiv C_1 \wedge C_2$. Then $\mu \supseteq T_P \uparrow \alpha(F\theta) = T_P \uparrow \alpha(C_1\theta) \otimes T_P \uparrow \alpha(C_2\theta) = \mu_1 \otimes \mu_2$. By the induction hypothesis, there exists a clause in $NF(P)$ having an instance of the form $C_1' : \lambda_0 \leftarrow F_1 : \lambda_1 \wedge \cdots \wedge F_n : \lambda_n$ such that $\lambda_0 \subseteq \mu_1$ and for $1 \leqslant i \leqslant n$, $T_P \uparrow (\alpha - 1)(F_i) \subseteq \lambda_i$ and C_1' subsumes $C_1\theta$. Similarly, there exists a clause in $NF(P)$ having an instance of the form $C_2' : \rho_0 \leftarrow G_1 : \rho_1 \wedge \cdots \wedge G_n : \rho_m$ such that $\rho_0 \subseteq \mu_2$ and for $1 \leqslant i \leqslant m$, $T_P \uparrow (\alpha - 1)(G_i) \subseteq \rho_i$ and C_2' subsumes $C_2\theta$. Then the clause $C \equiv (C_1 \wedge C_2)\,\theta : \lambda_0 \otimes \rho_0 \leftarrow F_1 : \lambda_1 \wedge \cdots \wedge F_n : \lambda_n \wedge G_1 : \rho_1 \wedge \cdots \wedge G_n : \rho_m$ is an instance of a clause in $NF(P)$. Moreover, $\lambda_0 \otimes \rho_0 \subseteq \mu_1 \otimes \mu_2 \subseteq \mu$. Thus the clause C satisfies the requirement of the lemma.

Case 2. $F \equiv D_1 \vee D_2$. The proof is similar to the one for conjunctions in case 1. This completes the induction and the proof of the lemma. ∎

We now demonstrate that SLDp-resolution is complete when we consider *consistent* p-programs.

THEOREM 8 (Completeness of SLDp-refutation). *Let P be a consistent p-program and Q be a query. Then if $P \models Q$, there exists an SLDp-refutation of Q from P.*

Proof. Let $Q \equiv \exists(F_1 : \mu_1 \wedge \cdots \wedge F_m : \mu_m)$. Then by Lemma 18, there is

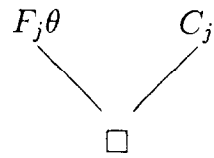

FIG. 1. Unrestricted refutation of $F_j\theta$ (base case).

a θ such that $Q\theta$ is ground and $P \models (Q\theta)$. Therefore, for all $1 \leqslant j \leqslant m$, $P \models (F_j\theta : \mu_j)$, where $F_j\theta$ is ground.

Case 1. $F_j \equiv A_1^j \wedge \cdots \wedge A_{n_j}^j$. Then by Lemma 16, for all $1 \leqslant i \leqslant n_j$, $P \models (A_i^j\theta : \rho_i)$ for some ρ_i such that $\rho_1 \otimes \cdots \otimes \rho_{n_j} \subseteq \mu_j$. Therefore, $lfp(T_P)(A_i^j\theta) \subseteq \rho_i$. Thus, there exists an integer $\alpha_i^j < \omega$ such that $T_P \uparrow \alpha_i^j(A_i^j\theta) \subseteq \rho_i$. Now pick $\alpha_j = max\{\alpha_i^j \mid 1 \leqslant i \leqslant n_j\}$. Since T_P is monotonic, $T_P \uparrow \alpha_j(A_i^j\theta) \subseteq \rho_i$, for all $1 \leqslant i \leqslant n_j$. Since $\otimes$ is monotonic, then for all $1 \leqslant j \leqslant m$, $T_P \uparrow \alpha_j(F_j\theta) = T_P \uparrow \alpha_j(A_1^j\theta) \otimes \cdots \otimes T_P \uparrow \alpha_j(A_{n_j}^j\theta) \subseteq \rho_1 \otimes \cdots \otimes \rho_{n_j} \subseteq \mu_j$.

Case 2. $F_j \equiv A_1^j \vee \cdots \vee A_{n_j}^j$. Then by Lemma 16, for all $1 \leqslant i \leqslant n_j$, $P \models (A_i^j\theta : \rho_i)$ for some ρ_i such that $\rho_1 \oplus \cdots \oplus \rho_{n_j} \subseteq \mu_j$. Therefore, $lfp(T_P)(A_i^j\theta) \subseteq \rho_i$. Thus, there exists an integer $\alpha_i^j < \omega$ such that $T_P \uparrow \alpha_i^j(A_i^j\theta) \subseteq \rho_i$. Now pick $\alpha_j = max\{\alpha_i^j \mid 1 \leqslant i \leqslant n_j\}$. Since T_P is monotonic, $T_P \uparrow \alpha_j(A_i^j\theta) \subseteq \rho_i$, for all $1 \leqslant i \leqslant n_j$. Since $\oplus$ is monotonic, then for all $1 \leqslant j \leqslant m$, $T_P \uparrow \alpha_j(F_j\theta) = T_P \uparrow \alpha_j(A_1^j\theta) \oplus \cdots \oplus T_P \uparrow \alpha_j(A_{n_j}^j\theta) \subseteq \rho_1 \oplus \cdots \oplus \rho_{n_j} \subseteq \mu_j$.

Therefore, by combining cases 1 and 2 above, it follows that for all basic formulas $F_1, \ldots, F_m$, there exists an integer α_j such that $T_P \uparrow \alpha_j(F_j\theta) \subseteq \mu_j$, for $1 \leqslant j \leqslant m$. Now pick $\alpha = max\{\alpha_j \mid 1 \leqslant j \leqslant m\}$. Since T_P is monotonic, $T_P \uparrow \alpha(F_j\theta) \subseteq \mu_j$, for all $1 \leqslant j \leqslant m$. Now we proceed by induction on α to prove that there exists an SLDp-refutation of $F_j\theta$ for all $1 \leqslant j \leqslant m$.

Base Case: $\alpha = 1$. By Lemma 19, there exists a clause in $NF(P)$ having a ground instance $C_j \equiv F_j\theta : \mu_j' \leftarrow$, where $\mu_j' \subseteq \mu_j$. Then Fig. 1 shows

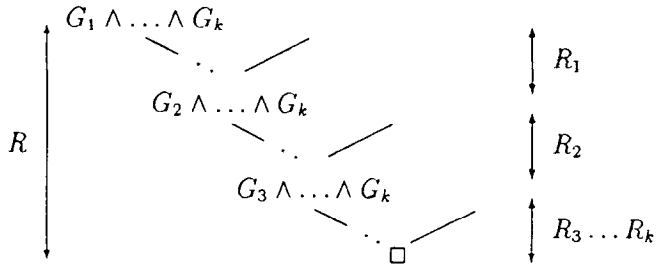

FIG. 2. SLDp-refutation of $G_1 : \lambda_1 \wedge \cdots \wedge G_k : \lambda_k$.

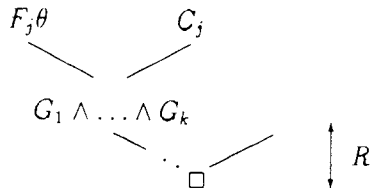

FIG. 3. Unrestricted refutation of $F_j\theta$ (inductive case).

an unrestricted refutation of $F_j\theta$. By the Max-gu Lemma, there exists an SLDp-refutation of $F_j\theta$.

Inductive Case: $\alpha > 1$. Either (i) $T_P \uparrow (\alpha - 1)(F_j\theta) \subseteq \mu_j$, in which case, by the induction hypothesis, there is an SLDp-refutation of $F_j\theta$, or

(ii) By Lemma 19, there exists a clause in $NF(P)$ having a ground instance $C_j \equiv F_j\theta : \mu_j' \leftarrow G_1 : \lambda_1 \wedge \cdots \wedge G_k : \lambda_k$ such that $\mu_j' \subseteq \mu_j$ and for all $1 \leqslant i \leqslant k$, $T_P \uparrow (\alpha - 1)(G_i) \subseteq \lambda_i$. By the induction hypothesis, each G_i has an SLDp-refutation R_i. Therefore, the ground query $G_1 : \lambda_1 \wedge \cdots \wedge G_k : \lambda_k$ has an SLDp-refutation R, as shown in Fig. 2. Hence Fig. 3 shows an unrestricted refutation of $F_j\theta$. By the Max-gu Lemma, $F_j\theta$ has an SLDp-refutation. This completes the induction on α.

Thus for each $1 \leqslant j \leqslant m$, there is an SLDp-refutation of $F_j\theta$. And these refutations can be combined into an SLDp-refutation of $Q\theta$, analogous to the refutation shown in Fig. 2. Finally, by the Lifting Lemma, there is an SLDp-refutation of Q. ▮

6. EXAMPLES AND DISCUSSION

We now present a few examples to show how SLDp-refutation works.

EXAMPLE 9. Let P be the simple p-program containing the clauses

$$p : [0.7, 0.7] \leftarrow q : [0.3, 0.4] \tag{1}$$

$$q : [0.3, 0.3] \leftarrow . \tag{2}$$

Consider now the query, $Q_1 \equiv p : [0.7, 0.7]$. A SLDp-refutation of this query is shown below:

$p : [0.7, 0.7]$	Initial Query	(3)
$q : [0.3, 0.4]$	Resolving 3, 1	(4)
$\square$	Resolving 4, 2.	(5)

Consider the more complex query $Q_2 \equiv (p \vee p) : [0.7, 0.8]$. Clearly, $P \models Q_2$. And $(p \vee p) : [0.7, 0.8]$ can resolve with clause (1) in $NF(P)$. Thus the remainder of the refutation proceeds in the same way as the refutation of Q_1.

In everyday reasoning, the occurrence of a single event E_1 may be the cause for some action. Likewise, event E_2 may be the cause for some other action. However, the simultaneous occurrence of events E_1 and E_2 may necessitate action that neither event, if occurring individually, would necessitate. For instance, a suspected murderer may be mildly alarmed when questioned by the police (event E_1). He may hire an attorney (action A_1). When he finds himself being followed by investigators (event E_2), he becomes more alarmed and takes action (action A_2) to ditch his shadows. Having ditched his shadows, he hears on the radio that his father is being questioned by the police (event E_3). He decides to flee to an obscure country (action A_3). Clearly, event E_1 is routine and should not lead to action A_3. Event E_3 by itself is also routine and should not necessitate A_3. However, events E_1, E_2, and E_3 combined may be the cause for drastic action (e.g., if the father can invalidate the murderer's alibi) and necessitate flight. Example 10 below presents another scenario.

EXAMPLE 10. Viper Labs is a small medical laboratory. It has facilities to conduct three kinds of tests (called t_1, t_2, and t_3) for identifying certain poisons secreted in the fangs of Indian vipers. There are three known species of vipers in India, and each secretes a different kind of poison (poisons p_1, p_2, and p_3). Viper poison acts rapidly on the human circulatory system, and if prompt action is not taken, a person bitten by a viper may die. Not much is known about the general properties of such vipers as they are rather elusive creatures. However, based on statistical data derived from previous experience with these creatures (and their hapless victims!), it has been concluded that the figures shown in the table below hold:

t_1	t_2	t_3	p_1	p_2	p_3
pos	*pos*	*pos*	95%	50%	75%
pos	*pos*	*neg*	80%	35%	15%
pos	*neg*	*pos*	85%	25%	—
pos	*neg*	*neg*	20%	—	15%
neg	*pos*	*pos*	75%	100%	50%
neg	*pos*	*neg*	40%	35%	—
neg	*neg*	*pos*	10%	25%	50%
neg	*neg*	*neg*	—	—	—

The entries in the above table that are not filled in are to be treated as "don't knows." Intuitively, the first row in the above table says that if an individual X who has been bitten by an Indian viper tests positive on all three tests, then there is a 95% chance that he was bitten by a viper secreting poison p_1. (We will assume that all the entries in the above table have a margin of error of $\pm 5\%$.) Thus, this really says that the probability that X is affected by poison p_1 lies in the 90–100% range.

The question is: Exactly how do we translate this table into a probabilistic logic program? We show two possibilities below.

Possibility 1. We could translate the first row into the clauses

$$p_1(X) : [0.9, 1] \leftarrow t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, pos) : [1, 1] \quad (1)$$

$$p_2(X) : [0.45, 0.55] \leftarrow t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, pos) : [1, 1] \quad (2)$$

$$p_3(X) : [0.7, 0.8] \leftarrow t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, pos) : [1, 1]. \quad (3)$$

Intuitively, the first clause says that if person X tests positive for t_1, t_2, and t_3, then the probability that X is affected by poison p_1 lies in the 90–100% range. The second clause says that if person X tests positive for t_1, t_2, and t_3, then the probability that X is affected by poison p_2 lies in the 45–55% range (and similarly for the third clause). Likewise, the second row in the table translates into the three clauses

$$p_1(X) : [0.75, 0.85] \leftarrow t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, neg) : [1, 1] \quad (4)$$

$$p_2(X) : [0.3, 0.4] \leftarrow t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, neg) : [1, 1] \quad (5)$$

$$p_3(X) : [0.1, 0.2] \leftarrow t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, neg) : [1, 1]. \quad (6)$$

In the same way, we can translate the other rows in the table and thus obtain a p-program. Let us call this program P. Let us assume that p also

contains three facts about a person called *joe*, viz., that *joe* tested positive on all three tests:

$$t_1(joe, pos) : [1, 1] \leftarrow \tag{7}$$

$$t_2(joe, pos) : [1, 1] \leftarrow \tag{8}$$

$$t_3(joe, pos) : [1, 1] \leftarrow . \tag{9}$$

Then it is verifiable that

$$P \models p_1(joe) : [0.9, 1]$$

and

$$P \models p_2(joe) : [0.45, 0.55]$$

and

$$P \models p_3(joe) : [0.7, 0.8].$$

A proof of $p_1(joe) : [0.9, 1]$ is shown below:

$$p_1(joe) : [0.9, 1] \quad \text{Initial Query} \tag{10}$$

$$t_1(X, pos) : [1, 1] \wedge t_2(X, pos) : [1, 1] \wedge t_3(X, pos) : [1, 1] \quad \text{Resolving 10, 1} \tag{11}$$

$$t_2(X, pos) : [1, 1] \wedge t_3(X, pos) : [1, 1] \quad \text{Resolving 11, 7} \tag{12}$$

$$t_3(X, pos) : [1, 1] \quad \text{Resolving 12, 8} \tag{13}$$

$$\square \quad \text{Resolving 13, 9} \tag{14}$$

Possibility 2. An alternative possibility is to translate the first row into the following three clauses:

$$p_1(X) : [0.9, 1] \leftarrow (t_1(X, pos) \wedge t_2(X, pos) \wedge t_3(X, pos)) : [1, 1] \tag{1}$$

$$p_2(X) : [0.45, 0.55] \leftarrow (t_1(X, pos) \wedge t_2(X, pos) \wedge t_3(X, pos)) : [1, 1] \tag{2}$$

$$p_3(X) : [0.7, 0.8] \leftarrow (t_1(X, pos) \wedge t_2(X, pos) \wedge t_3(X, pos)) : [1, 1]. \tag{3}$$

Here, the program clause bodies contain annotated conjunctions. Repeating this procedure for every row, we get a program Q which also contains the clause

$$(t_1(X, pos) \wedge t_2(X, pos) \wedge t_3(X, pos)) : [1, 1] \leftarrow . \tag{4}$$

It is easy to ascertain that $NF(Q)$ contains the clause

$$p_1(X) \vee p_3(X) : [0.9, 1] \leftarrow (t_1(X, pos) \wedge t_2(X, pos) \wedge t_3(X, pos)) : [1, 1]. \quad (5)$$

In addition, the following can be proved:

$$Q \models p_1(joe) : [0.9, 1]$$

and

$$Q \models p_2(joe) : [0.45, 0.55]$$

and

$$Q \models p_3(joe) : [0.7, 0.8].$$

Suppose now that we wish to find suitable medication for Joe. We would like to find the most appropriate medication to give him. Suppose we have three medicines m_1, m_2, m_3. We may have the rules

$$give(X, m_1, [1]) : [1, 1] \leftarrow p_1(X) : [0.7, 1] \quad (6)$$

$$give(X, m_2, [2]) : [1, 1] \leftarrow p_2(X) : [0.6, 1] \quad (7)$$

$$give(X, m_3, [1]) : [1, 1] \leftarrow p_1(X) : [0.6, 1] \quad (8)$$

$$give(X, m_1, [1, 2]) : [1, 1] \leftarrow (p_1(X) \vee p_2(X)) : [0.6, 1] \quad (9)$$

$$give(X, m_3, [1, 3]) : [1, 1] \leftarrow (p_1(X) \vee p_3(X)) : [0.8, 1] \quad (10)$$

The first rule here says that if there is at least a 70% chance that X was bitten by a viper secreting p_1, then we should surely give him medicine m_1. The third argument of *give* just tells us which poisons are treated by this medication. On the other hand, the last rule is more complex; it says that if the probability of X being bitten by a viper secreting one of the poisons p_1 or p_3 is greater than or equal to 80%, then we should give X medicine m_3. Let *VIPER* be the program Q together with the above rules defining *give*. Then

$$VIPER \models give(joe, m_1, [1]) : [1, 1]$$

$$VIPER \models give(joe, m_2, [2]) : [0, 1]$$

$$VIPER \models give(joe, m_3, [1, 3]) : [1, 1]$$

$$VIPER \models (p_1(X) \vee p_3(X)) : [0.9, 1].$$

A proof of $give(joe, m_3, [1, 3]) : [1, 1]$ is shown below:

$$give(joe, m_3, [1, 3]) : [1, 1] \quad \text{Initial Query} \quad (11)$$

$$(p_1(X) \vee p_3(X)) : [0.8, 1] \quad \text{Resolving 11, 10} \quad (12)$$

$$(t_1(X, pos) \wedge t_2(X, pos) \wedge t_3(X, pos)) : [1, 1] \quad \text{Resolving 12, 5} \quad (13)$$

$$\square \quad \text{Resolving 13, 4.} \quad (14)$$

A doctor may wish to use medicine m_3 to treat Joe as it treats both p_1 and p_3. Of course, if Joe was bitten by a snake secreting venom p_2, then he is going to die under this treatment. Such decisions are often made in critical situations, especially if medicine m_2 is incompatible with medicine m_3.

We now present a few simple examples to demonstrate the specialization of the probabilistic framework developed here to the case of two valued logic.

EXAMPLE 11 (Certain Information). Consider the following classical Horn program P':

$$p(X) \leftarrow q(X)$$
$$q(a) \quad \leftarrow .$$

A corresponding p-program P is

$$p(X) : [1, 1] \leftarrow q(X) : [1, 1]$$
$$q(a) : [1, 1] \leftarrow .$$

It is easy to verify that $lfp(T_P) = T_P \uparrow 2$, which is the following formula function:

$$T_P \uparrow 2(p(a)) = [1, 1],\ T_P \uparrow 2(q(a)) = [1, 1],$$
$$T_P \uparrow 2(p(a) \wedge q(a)) = [1, 1],$$
$$T_P \uparrow 2(p(a) \vee q(a)) = [1, 1].$$

Then $\mathscr{LP}(lfp(T_P))$ is the system of constraints

$$1 \leqslant (k_1 + k_2) \leqslant 1$$
$$1 \leqslant (k_1 + k_3) \leqslant 1$$
$$1 \leqslant k_1 \leqslant 1$$
$$1 \leqslant (k_1 + k_2 + k_3) \leqslant 1$$
$$k_1 + k_2 + k_3 + k_4 = 1,$$

and

$$k_1, k_2, k_3, k_4 \geqslant 0,$$

where the four "possible worlds" are $K_1 = \{p(a), q(a)\}$, $K_2 = \{p(a)\}$, $K_3 = \{q(a)\}$, and $K_4 = \varnothing$. But the only solution to $\mathscr{LP}(lfp(T_P))$ is $k_1 = 1$, $k_2 = k_3 = k_4 = 0$, indicating that $\{p(a), q(a)\}$ is the only "real world," and that there is a unique probabilistic model for P'. Thus, in this case, the probabilistic semantics coincides with the classical logic semantics.

The following result is easy to prove.

PROPOSITION 1. *Suppose P is any classical logic program (cf. Lloyd, 1987). Let Q be the p-program obtained by annotating all atoms in P with $[1, 1]$. Let I be any Herbrand interpretation in the sense of Lloyd (1987). Let $pr(I)$ be the atomic function that assigns $[1, 1]$ to all $A \in I$ and $[0, 0]$ to all $A \notin I$. Then, for all ground atoms A, $A \in T_P(I)$ iff $T_P(pr(I))(A) = [1, 1]$. (The first occurrence of T_P is the operator defined in Lloyd, 1987.)*

EXAMPLE 12. Consider the following classical Horn program P':

$$p(a) \leftarrow q(a).$$

A corresponding p-program P is

$$p(a) : [1, 1] \leftarrow q(a) : [1, 1].$$

Then $lfp(T_P) = T_P \uparrow 0 = \bot$, indicating that nothing can be concluded. The behaviour is the same if $q \equiv p$ or $q \equiv (\text{not } p)$.

EXAMPLE 13 (Inconsistent Information). Consider the following set P' of clauses:

$$p(a) \leftarrow q(a)$$
$$(\text{not } p(a)) \leftarrow q(a)$$
$$q(a).$$

A corresponding p-program P is

$$p(a) : [1, 1] \leftarrow q(a) : [1, 1]$$
$$p(a) : [0, 0] \leftarrow q(a) : [1, 1]$$
$$q(a) : [1, 1].$$

Then $lfp(T_P)(p(a)) = \varnothing$, indicating that program P is inconsistent. (It is easy to see that our notion of inconsistency generalizes the one used in the classical framework.)

As we have noted earlier, the semantics presented here is not paraconsistent in nature; in particular, inconsistent p-programs entail everything. The least fixpoint of T_P, however, may assign $\varnothing$ to some basic formulas, without assigning $\varnothing$ to all basic formulas. Hence, if we consider $lfp(T_P)$ as a "reasonable" way to interpret an inconsistent theory, then this semantics is paraconsistent. Whether one chooses to adopt this approach or not depends on the user. Consider the following example adapted from Bacchus (1988):

EXAMPLE 14. Suppose we know that over 80% of all dogs bark, and that Fido and Benjy are dogs. However, Benjy is unable to bark (his vocal cords were injured at some point). This can be represented as

$$bark(X) : [0.8, 1] \leftarrow dog(X) : [1, 1]$$
$$dog(fido) : [1, 1] \leftarrow$$
$$dog(benjy) : [1, 1] \leftarrow$$
$$bark(benjy) : [0, 0] \leftarrow .$$

Here, $lfp(T_P)$ assigns $\varnothing$ to $bark(benjy)$ and $[0.8, 1]$ to $bark(fido)$. Thus, as far as Benjy is concerned, this database contains some inconsistency, but the existence of this inconsistency does not affect Fido.

In general, our approach to representing probabilistic information is a subjectivistic view, while Bacchus' (1988) approach uses an explicitly empirical representation of probabilities. However, as noted by Bacchus (1988, p. 19–20), "the possible worlds approach, which expresses a subjective probability, can assign a probability to a closed formula, but is incapable of representing empirical probabilities." On the other hand, Bacchus' system "is incapable of representing a subjective probability assignment to a closed formula" (1988, p. 20). In addition, it is not clear how to use Bacchus' system as a basis for logic programming.

Note that the program in Example 14 can also be represented in the framework of Blair and Subrahmanian (1987) and Kifer and Subrahmanian (1992). According to their semantics, which is paraconsistent, this program has a model. This program also shows that when intervals are considered to be truth values (as is possible in Blair and Subrahmanian, 1987; Kifer and Subrahmanian, 1991), the resulting semantics is different from the probabilistic semantics.

The integration of logic and probability theory has been the subject

of numerous studies (Boole, 1854; Carnap, 1962; Hailperin, 1984; Lukasiewicz, 1970; Peirce, 1883; Dempster, 1968; Fagin and Halpern, 1988; Nilsson, 1986; Shafer, 1976). Here, we only compare our work with that which has been directly related to our efforts. Nilsson (1986) has given an informal operational account for integrating probabilities into logic. His framework lacks a model theory. Fagin, Halpern, and Megiddo (1989), in a pioneering paper, propose a model-theoretic basis for reasoning about systems of linear inequalities. Their aim is orthogonal to ours; they are concerned with satisfiability of such systems rather than with the assignment of *probabilistic* truth values to propositions. They explicitly state (Fagin *et al.*, 1989, p. 2) that in their system "All formulas are either true or false. They do not have probabilistic truth values." In this respect, our aims and techniques are different from those of Fagin *et al.* (1989). However, there seem to be a number of connections between our work and theirs; in particular, it would be interesting to see if their measure-theoretic approach could be used to develop a foundation for probabilistic logic programming, with inner measures serving as lower probability bounds, and outer measures serving as upper bounds for probabilities. We are currently studying this topic. In a related context, Kyburg (1974) has used a complex metalanguage that includes ZF set theory to express satistical information. Our proposal achieves similar goals within a first order framework. It is not clear how Kyburg's proposal may be used as the basis for a programming language (it was not intended for that).

The well-known Dempster–Shafer theory of evidence (Dempster, 1968; Shafer, 1976) does not seem to fit into our framework in any immediate way. There is some controversy, at present, on the epistemological basis for Dempster–Shafer theory. For instance, Cheeseman (1985) argues that the theory of Dempster–Shafer belief functions is ad hoc and non-probabilistic (cf. also Shafer, 1976). However, recent results of Fagin *et al.* (1989) indicate a closer connection between Dempster–Shafer theory and probability theory. We avoid this controversy and note that a great deal of work still needs to be done in the development of a model-theoretic foundation for Dempster–Shafer theory. Fitting (1988b) observes that developing quantitative logic programming languages based on Dempster–Shafer theory is still an open problem. Bandler and Kohout (1984) suggest an interval valued representation of multivalued logical operations. Their framework is based on fuzzy set theory, and they compute the lower and upper bounds of an interval with the use of min-max and product-sum. Fuzzy set theory (Zadeh, 1965, 1968) which also plays an important role in uncertain reasoning is well known to possess non-probabilistic features and hence we do not discuss it in greater detail here.

In logic programming, most work on quantitative deduction has focused on non-probabilistic logic programming. We feel that one reason for this

has been because the relationship between logic and probability theory has been elusive. The frameworks of Blair and Subrahmanian (1988) have dealt with lattice-based logic programming. A similar comment applies to the work of Fitting who interprets conjunction and disjunction as GLB and LUB, respectively, in the lattice. However, probabilities do not respect this interpretation (e.g., the probability of a disjunction may be much greater than the LUB of the probabilities of the individual disjuncts). van Emden (1986) develops a quantitative logic programming language in which multiplication is used to assign truth values to conjunctions. Of course, probabilities can be multiplied only if the events are independent, and hence van Emden's framework is also non-probabilistic. Baldwin (1987) develops an operational model for evidential logic programming which is based on fuzzy set theory, and there is no immediately forthcoming model-theoretic basis for his work.

Our framework has its limitations. In particular, there is no provision made for expressing conditional probabilities. In addition, we assume that programs are function free. This assumption was found to be necessary because we would like the set of inequalities determined by an atomic function to be finite. When function symbols are allowed, each ground atom (of which there may be an infinite number) determines an inequality, and hence we may have an infinite number of such inequalities. Solving an infinite set of inequations requires some different techniques. In a related context, Keisler (1985) has shown that even finite logics with σ-additive probability distributions (cf. Halmos, 1950) over the domain of an interpretation are not compact. We are currently working on this problem.

7. Conclusions

Thus far, quantitative logic programming languages (van Emden, 1986; Fitting, 1988a, b; Blair and Subrahmanian, 1987; Subrahmanian, 1987, 1990) have been unable to deal with probabilistic information. As probabilistic and statistical information is widely used in everyday decision-making, it is essential that logic programs have the ability to represent probabilistic information. We have proposed, in this paper, a probabilistic framework for logic programming. We have developed a probabilistic model theory and showed various connections between families of probabilistic models and the fixpoints of an operator associated with the program. Our probabilistic model theory satisfies the four properties that Fenstad (1980) states as desiderata for a function to be considered probabilistic. In addition, we have developed a sound and complete proof procedure for such languages. To our knowledge, this is the first probabilistic semantics for quantitative logic programming.

In ongoing research, we are studying how this framework can be used to reason about queueing systems and for developing logic operating systems. The latter is greatly facilitated by the fact that mutual exclusion of events E_1, E_2 can be expressed as $(E_1 \wedge E_2) : [0, 0]$ even though the individual probabilities of events E_1 and E_2 may be non-zero. We are also studying the support of variables in annotations and non-monotonic negation.

ACKNOWLEDGMENTS

We thank Fahiem Bacchus, Howard Blair, Wiktor Marek, Michael Kifer, and Maarten van Emden for numerous fruitful discussions and constructive comments on the manuscript. V. S. S. also thanks the Office of Graduate Studies and Research of the University of Maryland for financial support in the summer of 1990. R. N. thanks Timos Sellis for financial support. The research was partially sponsored by the National Science Foundation under Grants IRI-8719458 and IRI-9109755, and by the Army Research Office under Grant Number DAAL-03-92-G-0225.

RECEIVED February 20, 1990; FINAL MANUSCRIPT RECEIVED December 17, 1990

REFERENCES

ANDERSON, E. J., AND NASH, P. (1987), "Linear Programming in Infinite-Dimensional Spaces: Theory and Applications," Wiley, New York.

BACCHUS, F. (1988), "Representing and Reasoning with Probabilistic Knowledge," Research Report CS-88-31, University of Waterloo.

BALDWIN, J. F. (1987), Evidential support logic programming, *J. Fuzzy Sets Systems* **24**, 1–26.

BANDLER, W., AND KOHOUT, L. J. (1984), Unified theory of multivalued logical operations in the light of the checklist paradigm, *IEEE Trans. Systems Man Cybernet.*

BLAIR, H. A., AND SUBRAHMANIAN, V. S. (1987), Paraconsistent logic programming, *Theoret. Comput. Sci.* **68**, 35–54, preliminary version *in* "Proceedings, 7th Conference on Foundations of Software Technology and Theoretical Computer Science," pp. 340–360, Lecture Notes in Computer Science, Vol. 287, Springer-Verlag, Berlin/New York.

BLAIR, H. A., AND SUBRAHMANIAN, V. S. (1988), Paraconsistent foundations for logic programming, *J. Non-Classical Logic* **5**, 45–73.

BOOLE, G. (1854), "The Laws of Thought," MacMillan & Co., London.

BOOLOS, G., AND JEFFREY, R. (1980), "Computability and Logic," Cambridge Univ. Press, New York/London.

CARNAP, R. (1962), "The Logical Foundations of Probability," 2nd ed., Univ. of Chicago Press, Chicago.

CHEESEMAN, P. (1985), In defense of probability, *in* "Proc. IJCAI-85," pp. 1002–1009.

DA COSTA, N. C. A., ABE, J. M., AND SUBRAHMANIAN, V. S. (1991), Remarks on annotated logic, *Z. Math. Logik Grundlag. Math.* **37**, 561–570.

DA COSTA, N. C. A., SUBRAHMANIAN, V. S., AND VAGO, C. (1991), The paraconsistent logics $P\mathcal{T}$, *Z. Math. Logik Grundlag. Math.* **37**, 139–148.

DEMPSTER, A. P. (1968), A generalization of Bayesian inference, *J. Roy. Statis. Soc. Ser. B* **30**, 205–247.

FAGIN, R., AND HALPERN, J. (1988), Uncertainty, belief and probability, *in* "Proceedings IJCAI-89," Morgan Kauffman.

FAGIN, R., HALPERN, J. Y., AND MEGIDDO, N. (1989), A logic for reasoning about probabilities, draft manuscript.

FENSTAD, J. E. (1980), The structure of probabilities defined on first-order languages, *in* "Studies in Inductive Logic and Probabilities" (R. C. Jeffrey, Ed.), Vol. 2, pp. 251–262, Univ. of California Press, Berkeley.

FITTING, M. C. (1985), A Kripke–Kleene semantics for logic programming, *J. Logic Programming* **4**, 295–312.

FITTING, M. C. (1988a), Logic programming on a topological bilattice, *Fund. Inform.* **11**, 209–218.

FITTING, M. C. (1991), Bilattices and the semantics of logic programming, *J. Logic Programming* **11**, 91–116.

FITTING, M. C. (1988c), Bilattices and the theory of truth, *J. Philos. Logic*, to appear.

FRECHET, M. (1935), Généralisations du théorème des probabilités totales, *Fund. Math.* **25**, 379–387.

GAIFMAN, H. (1964), Concerning measures in first order calculi, *Israel J. Math.* **2**, 1–17.

GNEDENKO, B. V., AND KHINCHIN, A. Y. (1962), "An Elementary Introduction to the Theory of Probability," Dover, New York.

HAILPERIN, T. (1984), "Probability Logic," *Notre Dame J. Formal Logic* **25**, 198–212.

HALMOS, P. (1950), "Measure Theory," Springer, Berlin/New York.

KARWAN, M. H., LOFTI, V., TELGEN, J., AND ZIONTS, S. (1983), "Redundancy in Mathematical Programming: A State of the Art Survey," Springer, Berlin/New York.

KEISLER, H. J. (1985), Probability quantifiers, *in* "Model Theoretic Logics" (J. Barwise and S. Feferman, Eds.), Springer, Berlin/New York.

KELLEY, J. L. (1955), "General Topology," Springer, New York.

KIFER, M., AND LI, A. (1988), On the semantics of rule-based expert systems with uncertainty, *in* "2nd Intl. Conf. on Database Theory, Bruges, Belgium" (M. Gyssens, J. Paredaens, and D. Van Gucht, Eds.), pp. 102–117, Lecture Notes in Computer Science, Vol. 326, Springer-Verlag, Berlin/New York.

KIFER, M., AND LOZINSKII, E. (1989a), RI: A logic for reasoning with inconsistency, *in* "4th Symposium on Logic in Computer Science, Asilomar, CA," pp. 253–262.

KIFER, M., AND LOZINSKII, E. L. (1989b), A logic for reasoning with inconsistency, submitted for publication.

KIFER, M., AND SUBRAHMANIAN, V. S. (1992), Theory of generalized annotated logic programming and its applications, *J. Logic Programming* **12**, 335–367.

KOLMOGOROV, A. N. (1956), "Foundations of the Theory of Probability," Chelsea, New York.

KYBURG, H. (1974), "The Logic Foundations of Statistical Inference," Reidel, Dordrecht.

LASSEZ, J.-L., HUYNH, T., AND MCALOON, K. (1989), Simplification and elimination of redundant linear arithmetic constraints, *in* "Proceedings, 1989 North American Conference on Logic Programming" (E. Lusk and R. Overbeek, Eds.), pp. 37–51, MIT Press, Cambridge, MA.

LLOYD, J. W. (1987), "Foundations of Logic Programming," Springer, Berlin/New York.

LUKASIEWICZ, J. (1970), Logical foundations of probability theory, *in* "Selected Works of Jan Lukasiewicz" (L. Berkowski, Ed.), pp. 16–43, North-Holland, Amsterdam.

MARTELLI, A., AND MONTANARI, U. (1982), An efficient unification algorithm, *ACM Trans. Programming Languages Systems* **4**, 258–282.

MORISHITA, S. (1989), "A Unified Approach to Semantics of Multi-Valued Logic Programs," Technical Report RT 5006, IBM Tokyo.

NG, R. T., AND SUBRAHMANIAN, V. S. (1989), "Probabilistic Logic Programming," Technical Report, University of Maryland.

NILSSON, N. (1986), Probabilistic logic, *AI J.* **28**, 71–87.

PEIRCE, C. S. (1983), A theory of probable inference, *in* "Studies in Logic," pp. 126–181, Little, Brown, Boston.

SCHRIJVER, A. (1986), "Theory of Linear and Integer Programming," Wiley, New York.

SCOTT, D. S., AND KRAUSS, P. (1966), Assigning probabilities to logical formulas, *in* "Aspects of Inductive Logic" (J. Hintikka and P. Suppes, Eds.), North-Holland, Amsterdam.

SHAFER, G. (1976), "A Mathematical Theory of Evidence," Princeton Univ. Press, Princeton, NJ.

SHAPIRO, E. (1983), Logic programs with uncertainties: A tool for implementing expert systems, *in* "Proceedings, IJCAI 1983," pp. 529–532, William Kauffman.

SCHOENFIELD, J. (1967), "Mathematical Logic," Addison–Wesley, Reading, MA.

SUBRAHMANIAN, V. S. (1987), On the semantics of quantitative logic programs, *in* "Proceedings, 4th IEEE Symposium on Logic Programming," pp. 173–182, Computer Society Press, Washington, DC.

SUBRAHMANIAN, V. S. (1990), Mechanical proof procedures for many valued lattice based logic programming, *J. Non-Classical Logic* **7**, 7–41.

SUBRAHMANIAN, V. S. (1992), Paraconsistent disjunctive deductive databases, *Theoret. Comput. Sci.* **93**, 115–141.

VAN EMDEN, M. H. (1986), Quantitative deduction and its fixpoint theory, *J. Logic Programming* **4**, 37–53.

ZADEH, L. A. (1965), Fuzzy sets, *Inform. and Control* **8**, 338–353.

ZADEH, L. A. (1968), Fuzzy algorithms, *Inform. and Control* **12**, 94–102.